ALGORITMA DAN STUKTUR DATA

Doubly Linked List

Oleh :

Fina Salsabila Pramudita (5223600006)

Program Studi Sarjana Terapan Teknologi Game

Departemen Teknologi Multimedia Kreatif

Politeknik Elektronika Negeri Surabaya

2024

Doubly linked list dibentuk dengan Menyusun sejumlah elemen sehingga pointer next menunjuk ke elemen yang mengikutinya dan pointer back menunjuk ke elemen yang mendahuluinya. Doubly linked list terdiri dari elemen – elemen individu, Dimana masing – masing dihubungkan dengan dua pointer. Masing – masing elemen terdiri dari 3 bagian, yaitu sebuah data dan sebuah pointer yang berisi alamat data berikutnya disebut dengan next dan pointer yang berisi alamat data sebelumnya di sebut before.

PERCOBAAN

1. Implementasikan operasi dasar Double linked list : Menyisipkan sebagai simpul ujung(awal) dari linked list.
1. Implementasikan operasi dasar Double linked list : Membaca atau menampilkan
2. Implementasikan operasi dasar Double linked list : Mencari sebuah simpul tertentu. Tambahkan kondisi jika yang dicari adalah data yang paling depan.
3. Implementasikan operasi dasar Double linked list : Menghapus simpul tertentu.
4. Tambahkan kondisi jika yang dihapus adalah data yang paling depan atau data yang paling terakhir.
5. Gabungkan semua operasi di atas dalam sebuah Menu Pilihan.

Full Script

```
#include <iostream>
using namespace std;

//Mendefinisi Struktur Linked List
struct Node {
    int data;
    Node* prev;
    Node* next;
};

//Menyisipkan simpul baru di awal
void insertNode(Node*& head, Node*& tail, int data) {
    Node* n = new Node();
    n->data = data;

    if (head == nullptr) { //LinkedList masih kosong
        n->next = nullptr;
```

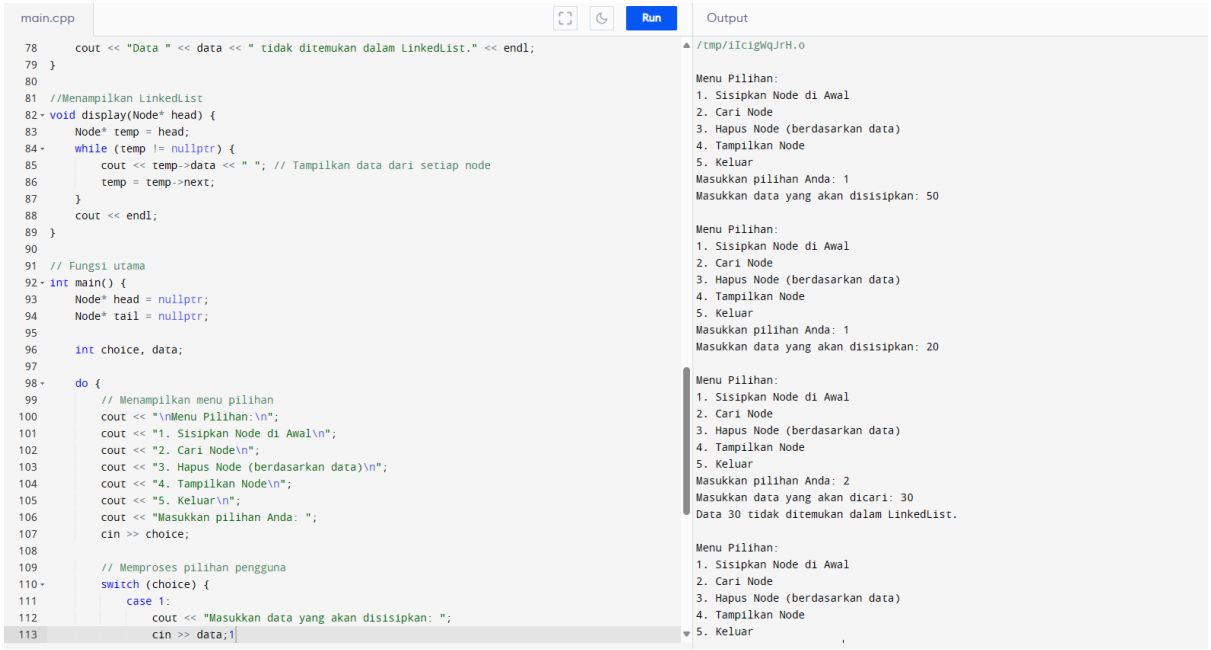
```
        n->prev = nullptr;
        head = n;
        tail = n;
    } else { //LinkedList sudah berisi node
        n->next = head;
        head->prev = n;
        head = n;
    }
}

//Mencari node dengan nilai tertentu dalam LinkedList
void search(Node* head, int data) {
    Node* temp = head;
    int position = 1;
    bool found = false;

    // Traverse LinkedList
    while (temp != nullptr) {
        if (temp->data == data) { // Jika data ditemukan
            found = true;
            break;
        }
        temp = temp->next;
        position++;
    }

    //Cek apakah data ditemukan
    if (found)
        cout << "Data " << data << " ditemukan pada node ke-" << position <<
endl;
    else
        cout << "Data " << data << " tidak ditemukan dalam LinkedList." <<
endl;
}

//Menghapus node dengan nilai tertentu
void deleteNode(Node*& head, Node*& tail, int data) {
    Node* temp = head;
```

```
    //mencari node dengan data yang sesuai
    while (temp != nullptr) {
        if (temp->data == data) {
            if (temp == head) {
                head = head->next;
                if (head != nullptr)
                    head->prev = nullptr;
                else // Jika node yang dihapus adalah satu-satunya node dalam
LinkedList
                    tail = nullptr;
            } else if (temp == tail) {
                tail = tail->prev;
                tail->next = nullptr;
            } else { // Jika node yang akan dihapus berada di tengah
LinkedList
                temp->prev->next = temp->next;
                temp->next->prev = temp->prev;
            }
            delete temp; //Hapus node
            return;
        }
        temp = temp->next;
    }

    // Jika data tidak ditemukan dalam LinkedList
    cout << "Data " << data << " tidak ditemukan dalam LinkedList." << endl;
}

//Menampilkan LinkedList
void display(Node* head) {
    Node* temp = head;
    while (temp != nullptr) {
        cout << temp->data << " "; // Tampilkan data dari setiap node
        temp = temp->next;
    }
    cout << endl;
}

//Isi Menu Pilihan
```

```
int main() {
    Node* head = nullptr;
    Node* tail = nullptr;

    int choice, data;

    do {
        // Menampilkan menu pilihan
        cout << "\nMenu Pilihan:\n";
        cout << "1. Sisipkan Node di Awal\n";
        cout << "2. Cari Node\n";
        cout << "3. Hapus Node (berdasarkan data)\n";
        cout << "4. Tampilkan Node\n";
        cout << "5. Keluar\n";
        cout << "Masukkan pilihan Anda: ";
        cin >> choice;

        // Memproses pilihan pengguna
        switch (choice) {
            case 1:
                cout << "Masukkan data yang akan disisipkan: ";
                cin >> data;
                insertNode(head, tail, data);
                break;
            case 2:
                cout << "Masukkan data yang akan dicari: ";
                cin >> data;
                search(head, data);
                break;
            case 3:
                cout << "Masukkan data yang akan dihapus: ";
                cin >> data;
                deleteNode(head, tail, data);
                break;
            case 4:
                cout << "LinkedList: ";
                display(head);
                break;
            case 5:
```

```
                cout << "Keluar dari program.\n";
                break;
            default:
                cout << "Pilihan tidak valid! Silakan coba lagi.\n";
        }
    } while (choice != 5);

    return 0;
}
```

HASIL PERCOBAAN



main.cpp

```
78      cout << "Data " << data << " tidak ditemukan dalam LinkedList." << endl;
79  }
80
81  //Menampilkan LinkedList
82  void display(Node* head) {
83      Node* temp = head;
84      while (temp != nullptr) {
85          cout << temp->data << " "; // Tampilkan data dari setiap node
86          temp = temp->next;
87      }
88      cout << endl;
89  }
90
91  // Fungsi utama
92  int main() {
93      Node* head = nullptr;
94      Node* tail = nullptr;
95
96      int choice, data;
97
98      do {
99          // Menampilkan menu pilihan
100         cout << "\nMenu Pilihan:\n";
101         cout << "1. Sisipkan Node di Awal\n";
102         cout << "2. Cari Node\n";
103         cout << "3. Hapus Node (berdasarkan data)\n";
104         cout << "4. Tampilkan Node\n";
105         cout << "5. Keluar\n";
106         cout << "Masukkan pilihan Anda: ";
107         cin >> choice;
108
109         // Memproses pilihan pengguna
110         switch (choice) {
111             case 1:
112                 cout << "Masukkan data yang akan disisipkan: ";
113                 cin >> data;1
```

Output

```
/tmp/iIcigWqJrH.o

Menu Pilihan:
1. Sisipkan Node di Awal
2. Cari Node
3. Hapus Node (berdasarkan data)
4. Tampilkan Node
5. Keluar
Masukkan pilihan Anda: 1
Masukkan data yang akan disisipkan: 50

Menu Pilihan:
1. Sisipkan Node di Awal
2. Cari Node
3. Hapus Node (berdasarkan data)
4. Tampilkan Node
5. Keluar
Masukkan pilihan Anda: 1
Masukkan data yang akan disisipkan: 20

Menu Pilihan:
1. Sisipkan Node di Awal
2. Cari Node
3. Hapus Node (berdasarkan data)
4. Tampilkan Node
5. Keluar
Masukkan pilihan Anda: 2
Masukkan data yang akan dicari: 30
Data 30 tidak ditemukan dalam LinkedList.

Menu Pilihan:
1. Sisipkan Node di Awal
2. Cari Node
3. Hapus Node (berdasarkan data)
4. Tampilkan Node
5. Keluar
```