

Laporan Praktikum

Algoritma dan Struktur Data

Ganjil 2023/2024
Program Studi Teknik Informatika
Institut Teknologi Sumatera



Modul : Single LinkedList

Nama : Aulia Putri Sayidina

NIM : 122140060

Kelas (Kelas Asal) : RC

Instruksi sederhana :

- Disarankan kepada **Praktikan Algoritma Struktur Data** untuk mengeditnya menggunakan Google Docs agar tidak berantakan dan rapi,
- Silahkan mengganti **Nama Modul** baik yang ada pada **Cover** dan **Header** sesuai dengan materi praktikum,
- Gunakan text styling seperti **Heading 1**, **Normal Text** yang telah terformat / Text Style lainnya yang digunakan untuk menjaga estetika laporan,
- Gunakan **Syntax Highlighter** untuk merapikan kode yang sudah Praktikan buat ke dalam Laporan Praktikum.

Materi Praktikum

PENDAHULUAN:

Dalam struktur data, Linked List adalah suatu struktur yang terdiri dari elemen-elemen data yang disusun dalam urutan linear. Setiap elemen dalam Linked List disebut sebagai "node," dan setiap node memiliki dua bagian utama: data dan pointer yang menunjuk ke node berikutnya. Salah satu jenis Linked List yang umum digunakan adalah Single Linked List.

FUNGSI LINKED LIST:

Linked List memberikan fleksibilitas dalam menyimpan dan mengelola data, terutama ketika ukuran data tidak diketahui pada awalnya atau membutuhkan penambahan dan penghapusan data secara dinamis.

Single linkedlist terdiri dari value dan pointer next.

Link repl.it Source Code

<https://onlinegdb.com/9ixiwBVs9>

Source Code

```
1. //Aulia Putri Sayidina
2. //122140060
3.
4. #include <iostream>
5. using namespace std;
6.
7. // Struktur untuk node
8. struct Node {
9.     int data;
10.    Node* next;
11.};
12.
13.// Fungsi untuk membuat node baru
14.Node* createNode(int value) {
15.    Node* newNode = new Node;
16.    newNode->data = value;
```

Praktikum Algoritma dan Struktur Data Ke-VI — Single LinkedList

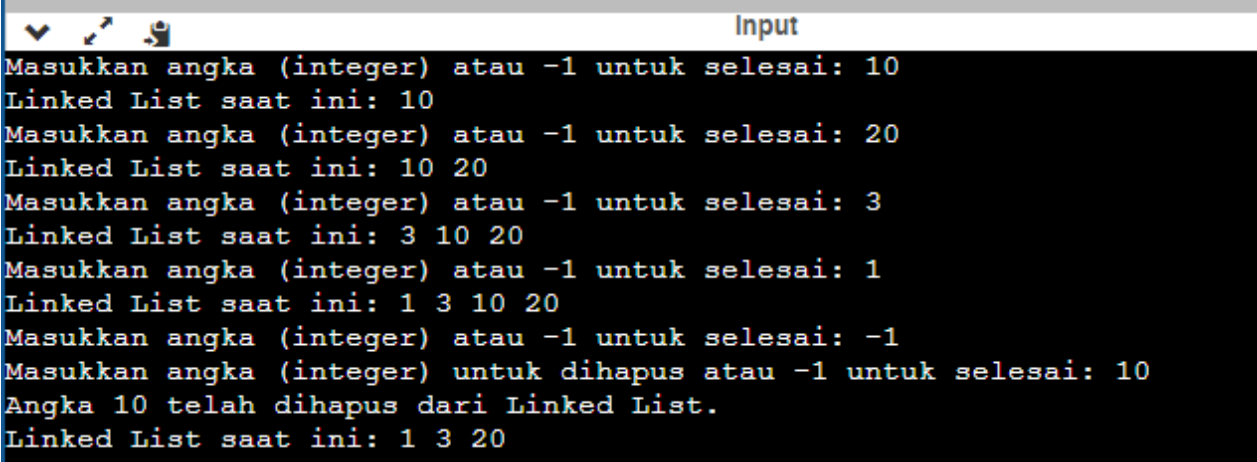
```
17.     newNode->next = nullptr;
18.     return newNode;
19. }
20.
21. // Fungsi untuk menambahkan node ke dalam linked list terurut
22. void insertSorted(Node*& head, int value) {
23.     Node* newNode = createNode(value);
24.
25.     // Kasus jika linked list masih kosong atau nilai baru lebih kecil dari head
26.     if (head == nullptr || value < head->data) {
27.         newNode->next = head;
28.         head = newNode;
29.         return;
30.     }
31.
32.     // Menemukan tempat yang tepat untuk menyisipkan node
33.     Node* current = head;
34.     while (current->next != nullptr && current->next->data < value) {
35.         current = current->next;
36.     }
37.
38.     // Menyisipkan node baru setelah current
39.     newNode->next = current->next;
40.     current->next = newNode;
41. }
42.
43. // Fungsi untuk menampilkan isi linked list
44. void displayList(Node* head) {
45.     Node* current = head;
46.     while (current != nullptr) {
47.         cout << current->data << " ";
48.         current = current->next;
49.     }
50.     cout << endl;
51. }
52.
53. // Fungsi untuk menghapus node pertama
54. void deleteFirst(Node*& head) {
55.     if (head == nullptr) {
56.         return;
57.     }
58.     Node* temp = head;
59.     head = head->next;
60.     delete temp;
61. }
```

```
62.
63.// Fungsi untuk menghapus node setelah node tertentu
64.void deleteAfter(Node* node) {
65.    if (node == nullptr || node->next == nullptr) {
66.        return;
67.    }
68.    Node* temp = node->next;
69.    node->next = temp->next;
70.    delete temp;
71.}
72.
73.// Fungsi untuk mencari dan menghapus node berdasarkan nilai
74.void deleteNode(Node*& head, int value) {
75.    Node* current = head;
76.    Node* previous = nullptr;
77.
78.    // Mencari node dengan nilai yang sesuai
79.    while (current != nullptr && current->data != value) {
80.        previous = current;
81.        current = current->next;
82.    }
83.
84.    // Jika node ditemukan
85.    if (current != nullptr) {
86.        // Jika node adalah node pertama
87.        if (previous == nullptr) {
88.            deleteFirst(head);
89.        } else {
90.            deleteAfter(previous);
91.        }
92.        cout << "Angka " << value << " telah dihapus dari Linked List." << endl;
93.    } else {
94.        cout << "Angka " << value << " tidak ditemukan dalam Linked List." << endl;
95.    }
96.}
97.
98.int main() {
99.    Node* head = nullptr;
100.    int input;
101.
102.    cout << "Masukkan angka (integer) atau -1 untuk selesai: ";
103.
104.    while (true) {
105.        cin >> input;
106.    }
```

Praktikum Algoritma dan Struktur Data Ke-VI — Single LinkedList

```
107.          // Selesai jika input -1
108.          if (input == -1) {
109.              break;
110.          }
111.
112.          // Memasukkan data ke dalam linked list secara terurut
113.          insertSorted(head, input);
114.
115.          // Menampilkan isi linked list
116.          cout << "Linked List saat ini: ";
117.          displayList(head);
118.
119.          cout << "Masukkan angka (integer) atau -1 untuk selesai: ";
120.      }
121.
122.      cout << "Masukkan angka (integer) untuk dihapus atau -1 untuk selesai:
123.      ";
124.      cin >> input;
125.      if(input == -1) {
126.          return 0;
127.      }
128.
129.      // Menghapus angka dari linked list
130.      deleteNode(head, input);
131.
132.      // Menampilkan isi linked list setelah penghapusan
133.      cout << "Linked List saat ini: ";
134.      displayList(head);
135.
136.      return 0;
137.  }
138.
```

Dokumentasi Hasil Running



```
Input
Masukkan angka (integer) atau -1 untuk selesai: 10
Linked List saat ini: 10
Masukkan angka (integer) atau -1 untuk selesai: 20
Linked List saat ini: 10 20
Masukkan angka (integer) atau -1 untuk selesai: 3
Linked List saat ini: 3 10 20
Masukkan angka (integer) atau -1 untuk selesai: 1
Linked List saat ini: 1 3 10 20
Masukkan angka (integer) atau -1 untuk selesai: -1
Masukkan angka (integer) untuk dihapus atau -1 untuk selesai: 10
Angka 10 telah dihapus dari Linked List.
Linked List saat ini: 1 3 20
```

Gambar 1. Hasil Running Code Tugas Single LinkedList

Referensi

Modul perkuliahan Algoritma Struktur Data – Single LinkedList dari ITERA