

**LAPORAN PRAKTIKUM  
STRUKTUR DATA**

**MODUL IV  
SINGLE LINKED LIST**



**Disusun Oleh :**

NAMA : Taufik Hafit Zakaria

NIM : 103112430093

**Dosen**

WAHYU ANDI SAPUTRA

**PROGRAM STUDI STRUKTUR DATA  
FAKULTAS INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO  
2025**

## A. Dasar Teori

Single Linked List merupakan salah satu bentuk struktur data dinamis yang tersusun atas kumpulan node (simpul) yang saling terhubung secara sekuensial. Setiap node terdiri dari dua bagian utama, yaitu data yang menyimpan informasi dan pointer (next) yang menunjuk ke node berikutnya. Berbeda dengan array yang bersifat statis, Single Linked List bersifat fleksibel karena ukuran list dapat berubah secara dinamis selama program berjalan tanpa perlu mendeklarasikan ukuran awal.

### Karakteristik Single Linked List

- Sifat Dinamis: Alokasi memori dilakukan saat program berjalan (runtime), sehingga mudah untuk menambah atau menghapus elemen tanpa perlu menggeser data lainnya.
- Satu Arah: Pointer pada setiap node hanya menunjuk ke node berikutnya, sehingga penelusuran data hanya dapat dilakukan dari node pertama (head) menuju akhir list (NULL).
- Node Terakhir: Node terakhir selalu memiliki pointer yang bernilai NULL menandakan akhir dari list.
- Operasi Dasar: Meliputi pembuatan list (create list), penyisipan elemen (insert), penghapusan elemen (delete), pencarian data (searching), dan pembaruan data (update).

### Keuntungan Single Linked List

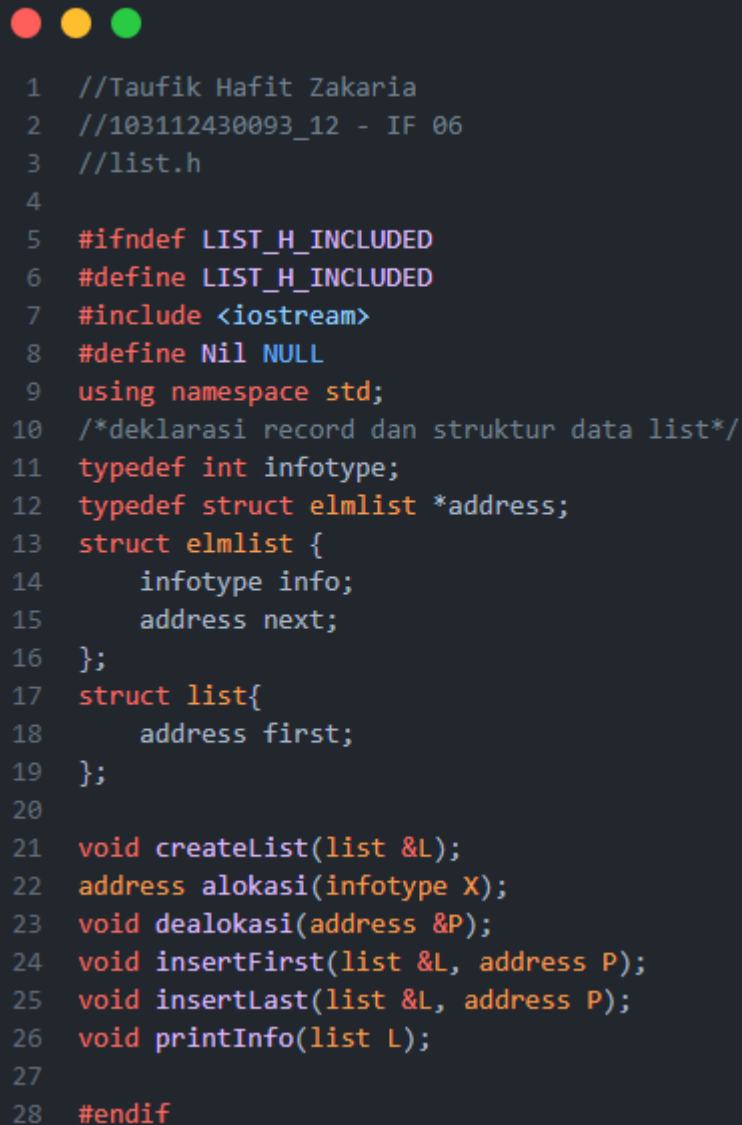
- Fleksibel: Dapat menambah atau menghapus elemen tanpa perlu menggeser elemen lain seperti pada array.
- Efisien dalam Memori: Memori dialokasikan sesuai kebutuhan data, sehingga lebih efisien dibanding array yang berukuran tetap.
- Struktur Dapat Dikembangkan: Menjadi dasar bagi struktur data lain seperti stack, queue, tree, dan graph.

### Keterbatasan Single Linked List

- Akses Tidak Langsung: Untuk mengakses elemen tertentu, pencarian harus dimulai dari node pertama karena tidak mendukung akses acak.
- Overhead Memori: Setiap node membutuhkan ruang tambahan untuk menyimpan pointer.
- Traversal Satu Arah: Tidak dapat menelusuri kembali ke node sebelumnya karena hanya memiliki satu pointer pengarah ke depan.

B. Guided (berisi screenshot source code & output program disertai penjelasannya)

Guided 1



```
1 //Taufik Hafit Zakaria
2 //103112430093_12 - IF 06
3 //list.h
4
5 #ifndef LIST_H_INCLUDED
6 #define LIST_H_INCLUDED
7 #include <iostream>
8 #define Nil NULL
9 using namespace std;
10 /*deklarasi record dan struktur data list*/
11 typedef int infotype;
12 typedef struct elmlist *address;
13 struct elmlist {
14     infotype info;
15     address next;
16 };
17 struct list{
18     address first;
19 };
20
21 void createList(list &L);
22 address alokasi(infotype X);
23 void dealokasi(address &P);
24 void insertFirst(list &L, address P);
25 void insertLast(list &L, address P);
26 void printInfo(list L);
27
28 #endif
```



```
1 //Taufik Hafit Zakaria
2 //103112430093_12 - IF 06
3 //singlylist.cpp
4
5 #include "singlylist.h"
6
7 // Membuat sebuah list baru yang masih kosong
8 void createList(list &L){
9     L.first = Nil;
10 }
11
12 // Mengalokasikan memori untuk sebuah elemen baru
13 // dan mengisi elemen tersebut dengan data (X)
14 address alokasi(infotype X){
15     address P = new elmlist;
16     P->info = X;
17     P->next = Nil;
18     return P;
19 }
20
21 // Menghapus atau membebaskan memori dari sebuah elemen
22 void dealokasi(address &P){
23     delete P;
24 }
25
26 // Menambahkan sebuah elemen baru di awal list
27 void insertFirst(list &L, address P){
28     P->next = L.first;
29     L.first = P;
30 }
```

```

31
32 // Menambahkan sebuah elemen baru di akhir list
33 void insertLast(list &L, address P){
34     if(L.first == Nil){
35         // Jika list masih kosong, elemen baru menjadi elemen pertama
36         insertFirst(L, P);
37     } else {
38         // Jika list sudah ada isinya, cari elemen terakhir
39         address Last = L.first;
40         while (Last->next != Nil){
41             Last = Last->next;
42         }
43         // Sambungkan elemen terakhir dengan elemen baru
44         Last->next = P;
45     }
46 }
47
48 // Menampilkan semua isi data dari setiap elemen di list
49 void printInfo(list L){
50     address P = L.first;
51     if (P == Nil) {
52         cout << "List kosong" << endl;
53     } else {
54         while (P != Nil) {
55             cout << P->info << " ";
56             P = P->next;
57         }
58         cout << endl;
59     }
60 }
61

```

```
1 //Taufik Hafit Zakaria
2 //103112430093_12 - IF 06
3 //main.cpp
4
5 #include <iostream>
6 #include <cstdlib>
7 #include "singlylist.h"
8 #include "singlylist.cpp"
9
10 using namespace std;
11
12 // Fungsi utama program
13 int main() {
14     // Deklarasi variabel list L dan address P
15     list L;
16     address P;
17
18     // 1. Membuat list kosong
19     createList(L);
20     cout << "membuat list menggunakan insertLast..."<<endl;
21
22     // 2. Mengisi list dengan beberapa elemen menggunakan insertLast
23     // Alokasi memori dan masukkan nilai 9 ke akhir list
24     P = alokasi(9);
25     insertLast(L, P);
26     // Alokasi memori dan masukkan nilai 12 ke akhir list
27     P = alokasi(12);
28     insertLast(L, P);
29     // Alokasi memori dan masukkan nilai 8 ke akhir list
30     P = alokasi(8);
31     insertLast(L, P);
32     // Alokasi memori dan masukkan nilai 0 ke akhir list
33     P = alokasi(0);
34     insertLast(L, P);
35     // Alokasi memori dan masukkan nilai 2 ke akhir list
36     P = alokasi(2);
37     insertLast(L, P);
38
39     // 3. Menampilkan seluruh isi list
40     cout << "isi list sekarang adalah: ";
41     printInfo(L);
42
43     // Menjeda program sebelum ditutup (khusus Windows)
44     system("pause");
45     return 0;
```

## Screenshots Output

```
● PS C:\programming\Struktur Data\LAPRAK 4\unguided> cd "c:\progra"
ming\Struktur Data\LAPRAK 4\guided\" ; if ($?) { g++ main.cpp -
o main } ; if ($?) { .\main }
membuat list menggunakan insertiLast...
isi list sekarang adalah: 9 12 8 0 2
Press any key to continue . . .
○ PS C:\programming\Struktur Data\LAPRAK 4\guided> |
```

## Deskripsi:

Program ini merupakan implementasi dari struktur data Single Linked List menggunakan bahasa C++. Single Linked List adalah struktur data dinamis yang tersusun atas simpul (node) yang saling terhubung, di mana setiap simpul memiliki dua komponen utama yaitu data (info) untuk menyimpan nilai dan pointer (next) yang menunjuk ke node berikutnya. Berbeda dengan array yang bersifat statis, struktur ini memungkinkan penambahan dan penghapusan elemen secara fleksibel tanpa perlu menggeser data lain. Dalam program ini terdapat tiga file utama, yaitu list.h yang berisi deklarasi struktur data dan fungsi, singlylist.cpp yang berisi implementasi fungsi, serta main.cpp yang menjalankan program utama. Fungsi-fungsi utama yang digunakan meliputi createList untuk membuat list kosong, alokasi dan dealokasi untuk mengatur memori, insertFirst dan insertLast untuk menambah elemen, serta printInfo untuk menampilkan isi list.

Program diawali dengan membuat list kosong, kemudian menambahkan beberapa elemen dengan nilai 9, 12, 8, 0, dan 2 menggunakan fungsi insertLast, sehingga setiap elemen baru ditambahkan di akhir list. Setelah seluruh data berhasil dimasukkan, fungsi printInfo digunakan untuk menampilkan isi list secara berurutan di layar. Hasil akhir yang ditampilkan menunjukkan data disusun sesuai urutan penyisipannya, yaitu 9 12 8 0 2. Program ini menggambarkan penerapan konsep pointer dan alokasi memori dinamis dalam pengelolaan data menggunakan Single Linked List, yang lebih efisien dibanding array karena dapat menyesuaikan ukuran list sesuai kebutuhan selama program berjalan.

C. Unguided/Tugas (berisi screenshot source code & output program disertai penjelasannya)

Unguided 1



```
1 //Taufik Hafit Zakaria
2 //103112430093_12 - IF 06
3 //playlist.h
4
5 #ifndef PLAYLIST_H_INCLUDED
6 #define PLAYLIST_H_INCLUDED
7
8 #include <iostream>
9 #include <string>
10 using namespace std;
11
12 struct Lagu {
13     string judul;
14     string penyanyi;
15     float durasi;
16     Lagu *next;
17 };
18
19 struct Playlist {
20     Lagu *head;
21 };
22
23 void createPlaylist(Playlist &P);
24 Lagu* createNode(string judul, string penyanyi, float durasi);
25 void tambahAwal(Playlist &P, Lagu *newNode);
26 void tambahAkhir(Playlist &P, Lagu *newNode);
27 void tambahSetelahKe3(Playlist &P, Lagu *newNode);
28 void hapusBerdasarkanJudul(Playlist &P, string judul);
29 void tampilkanPlaylist(Playlist P);
30
31 #endif
```



```
1 //Taufik Hafit Zakaria
2 //103112430093_12 - IF 06
3 //playlist.cpp
4
5 #include "Playlist.h"
6
7 void createPlaylist(Playlist &P) {
8     P.head = NULL;
9 }
10
11 Lagu* createNode(string judul, string penyanyi, float durasi) {
12     Lagu *newNode = new Lagu;
13     newNode->judul = judul;
14     newNode->penyanyi = penyanyi;
15     newNode->durasi = durasi;
16     newNode->next = NULL;
17     return newNode;
18 }
19
20 void tambahAwal(Playlist &P, Lagu *newNode) {
21     newNode->next = P.head;
22     P.head = newNode;
23 }
24
25 void tambahAkhir(Playlist &P, Lagu *newNode) {
26     if (P.head == NULL) {
27         P.head = newNode;
28     } else {
29         Lagu *temp = P.head;
30         while (temp->next != NULL) {
31             temp = temp->next;
32         }
33         temp->next = newNode;
34     }
35 }
```

```

36
37 void tambahSetelahKe3(Playlist &P, Lagu *newNode) {
38     if (P.head == NULL) {
39         cout << "Playlist masih kosong!" << endl;
40         return;
41     }
42
43     Lagu *temp = P.head;
44     int count = 1;
45     while (temp != NULL && count < 3) {
46         temp = temp->next;
47         count++;
48     }
49
50     if (temp == NULL) {
51         cout << "Playlist kurang dari 3 lagu!" << endl;
52     } else {
53         newNode->next = temp->next;
54         temp->next = newNode;
55         cout << "Lagu berhasil ditambahkan setelah lagu ke-3!" << endl;
56     }
57 }
58
59 void hapusBerdasarkanJudul(Playlist &P, string judul) {
60     if (P.head == NULL) {
61         cout << "Playlist kosong!" << endl;
62         return;
63     }
64
65     Lagu *temp = P.head;
66     Lagu *prev = NULL;
67
68     if (temp != NULL && temp->judul == judul) {
69         P.head = temp->next;
70         delete temp;
71         cout << "Lagu \"" << judul << "\" berhasil dihapus!" << endl;
72         return;
73     }

```

```

74
75     while (temp != NULL && temp->judul != judul) {
76         prev = temp;
77         temp = temp->next;
78     }
79
80     if (temp == NULL) {
81         cout << "Lagu dengan judul \"" << judul << "\" tidak ditemukan!" << endl;
82         return;
83     }
84
85     prev->next = temp->next;
86     delete temp;
87     cout << "Lagu \"" << judul << "\" berhasil dihapus!" << endl;
88 }
89
90 void tampilkanPlaylist(Playlist P) {
91     if (P.head == NULL) {
92         cout << "\nPlaylist kosong!\n";
93         return;
94     }
95
96     Lagu *temp = P.head;
97     int i = 1;
98     cout << "\n=== Daftar Playlist Lagu ===\n";
99     while (temp != NULL) {
100         cout << i << ". Judul   : " << temp->judul << endl;
101         cout << "    Penyanyi: " << temp->penyanyi << endl;
102         cout << "    Durasi   : " << temp->durasi << " menit" << endl << endl;
103         temp = temp->next;
104         i++;
105     }
106 }

```

```
1 //Taufik Hafit Zakaria
2 //103112430093_12 - IF 06
3 //playlist-main.cpp
4
5 #include <iostream>
6 #include "Playlist.h"
7 #include "Playlist.cpp"
8 using namespace std;
9
10 int main() {
11     Playlist myPlaylist;
12     createPlaylist(myPlaylist);
13
14     int pilihan;
15     string judul, penyanyi;
16     float durasi;
17
18     do {
19         cout << "\n===== MENU PLAYLIST LAGU =====" << endl;
20         cout << "1. Tambah lagu di awal playlist" << endl;
21         cout << "2. Tambah lagu di akhir playlist" << endl;
22         cout << "3. Tambah lagu setelah lagu ke-3" << endl;
23         cout << "4. Hapus lagu berdasarkan judul" << endl;
24         cout << "5. Tampilkan semua lagu" << endl;
25         cout << "0. Keluar" << endl;
26         cout << "Pilih menu: ";
27         cin >> pilihan;
28         cin.ignore();
```

```

29
30     switch (pilihan) {
31     case 1:
32         cout << "\nMasukkan judul lagu   : ";
33         getline(cin, judul);
34         cout << "Masukkan nama penyanyi: ";
35         getline(cin, penyanyi);
36         cout << "Masukkan durasi (menit): ";
37         cin >> durasi;
38         tambahAwal(myPlaylist, createNode(judul, penyanyi, durasi));
39         cout << "Lagu berhasil ditambahkan di awal!\n";
40         break;
41
42     case 2:
43         cout << "\nMasukkan judul lagu   : ";
44         getline(cin, judul);
45         cout << "Masukkan nama penyanyi: ";
46         getline(cin, penyanyi);
47         cout << "Masukkan durasi (menit): ";
48         cin >> durasi;
49         tambahAkhir(myPlaylist, createNode(judul, penyanyi, durasi));
50         cout << "Lagu berhasil ditambahkan di akhir!\n";
51         break;
52
53     case 3:
54         cout << "\nMasukkan judul lagu   : ";
55         getline(cin, judul);
56         cout << "Masukkan nama penyanyi: ";
57         getline(cin, penyanyi);
58         cout << "Masukkan durasi (menit): ";
59         cin >> durasi;
60         tambahSetelahKe3(myPlaylist, createNode(judul, penyanyi, durasi));
61         break;
62
63     case 4:
64         cout << "\nMasukkan judul lagu yang ingin dihapus: ";
65         getline(cin, judul);
66         hapusBerdasarkanJudul(myPlaylist, judul);
67         break;
68
69     case 5:
70         tampilkanPlaylist(myPlaylist);
71         break;
72
73     case 0:
74         cout << "\nKeluar dari program. Terima kasih!\n";
75         break;
76
77     default:
78         cout << "Pilihan tidak valid! Coba lagi.\n";
79     }
80 } while (pilihan != 0);
81
82 return 0;
83 }

```

## Screenshots Output

```
===== MENU PLAYLIST LAGU =====
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan semua lagu
0. Keluar
Pilih menu: 1

Masukkan judul lagu : tarot
Masukkan nama penyanyi: baskara
Masukkan durasi (menit): 3.20
Lagu berhasil ditambahkan di awal!
```

```
===== MENU PLAYLIST LAGU =====
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan semua lagu
0. Keluar
Pilih menu: 2

Masukkan judul lagu : kangen
Masukkan nama penyanyi: dewa
Masukkan durasi (menit): 4.30
Lagu berhasil ditambahkan di akhir!
```

```
===== MENU PLAYLIST LAGU =====
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan semua lagu
0. Keluar
Pilih menu: 3

Masukkan judul lagu : dan
Masukkan nama penyanyi: so7
Masukkan durasi (menit): 3.45
Playlist kurang dari 3 lagu!
```

```
===== MENU PLAYLIST LAGU =====
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan semua lagu
0. Keluar
Pilih menu: 1

Masukkan judul lagu   : sephia
Masukkan nama penyanyi: so7
Masukkan durasi (menit): 4.25
Lagu berhasil ditambahkan di awal!
```

```
===== MENU PLAYLIST LAGU =====
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan semua lagu
0. Keluar
Pilih menu: 5

=== Daftar Playlist Lagu ===
1. Judul   : sephia
   Penyanyi: so7
   Durasi  : 4.25 menit

2. Judul   : tarot
   Penyanyi: baskara
   Durasi  : 3.2 menit

3. Judul   : kangen
   Penyanyi: dewa
   Durasi  : 4.3 menit
```

```
===== MENU PLAYLIST LAGU =====
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan semua lagu
0. Keluar
Pilih menu: 3

Masukkan judul lagu   : film favorit
Masukkan nama penyanyi: so7
Masukkan durasi (menit): 4.45
Lagu berhasil ditambahkan setelah lagu ke-3!
```

```
===== MENU PLAYLIST LAGU =====
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan semua lagu
0. Keluar
Pilih menu: 5
```

```
=== Daftar Playlist Lagu ===
1. Judul   : sephia
   Penyanyi: so7
   Durasi  : 4.25 menit

2. Judul   : tarot
   Penyanyi: baskara
   Durasi  : 3.2 menit

3. Judul   : kangen
   Penyanyi: dewa
   Durasi  : 4.3 menit

4. Judul   : film favorit
   Penyanyi: so7
   Durasi  : 4.45 menit
```

```
===== MENU PLAYLIST LAGU =====
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan semua lagu
0. Keluar
Pilih menu: 4

Masukkan judul lagu yang ingin dihapus: tarot
Lagu "tarot" berhasil dihapus!
```



```
===== MENU PLAYLIST LAGU =====
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan semua lagu
0. Keluar
Pilih menu: 5
```

```
=== Daftar Playlist Lagu ===
1. Judul   : sephia
   Penyanyi: so7
   Durasi  : 4.25 menit

2. Judul   : kangen
   Penyanyi: dewa
   Durasi  : 4.3 menit

3. Judul   : film favorit
   Penyanyi: so7
   Durasi  : 4.45 menit
```

```
===== MENU PLAYLIST LAGU =====
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan semua lagu
0. Keluar
Pilih menu: 0

Keluar dari program. Terima kasih!
PS C:\programming\Struktur Data\LAPRAK 4\unguided>
```

#### Deskripsi:

Program ini merupakan implementasi struktur data Single Linked List dalam bentuk aplikasi manajemen playlist lagu menggunakan bahasa C++. Setiap lagu direpresentasikan sebagai sebuah node yang berisi data berupa judul lagu, nama penyanyi, dan durasi lagu, serta pointer next yang menunjuk ke lagu berikutnya. Struktur utama program terdiri dari dua komponen, yaitu Lagu sebagai node dan Playlist sebagai list yang menyimpan pointer ke lagu pertama (head). File playlist.h berisi deklarasi struktur data dan prototype fungsi, sedangkan playlist.cpp berisi implementasi fungsinya, seperti createPlaylist untuk membuat list kosong, createNode untuk membuat node baru, tambahAwal dan tambahAkhir untuk menambah lagu di awal atau akhir playlist, tambahSetelahKe3 untuk menambahkan lagu setelah lagu ketiga, hapusBerdasarkanJudul untuk menghapus lagu sesuai judulnya, dan tampilkanPlaylist untuk menampilkan seluruh lagu dalam playlist.

File `playlist-main.cpp` berperan sebagai program utama yang menampilkan menu interaktif agar pengguna dapat mengelola playlist dengan mudah. Pengguna dapat menambah lagu di awal, di akhir, atau setelah lagu ke-3, menghapus lagu berdasarkan judul, serta menampilkan seluruh daftar lagu beserta penyanyi dan durasinya. Program berjalan secara dinamis menggunakan pointer, sehingga setiap perubahan (penambahan atau penghapusan lagu) dilakukan tanpa perlu menggeser data lain di memori. Dengan demikian, program ini menggambarkan penerapan nyata dari konsep Single Linked List untuk mengelola data secara fleksibel dan efisien, khususnya dalam konteks penyimpanan dan manipulasi daftar lagu.

#### D. Kesimpulan

Dari hasil praktikum yang telah dilakukan, dapat disimpulkan bahwa struktur data Single Linked List memungkinkan penyimpanan dan pengelolaan data secara dinamis menggunakan pointer. Melalui implementasi program playlist lagu ini, konsep dasar operasi linked list seperti pembuatan list, penambahan elemen di awal dan akhir, penyisipan di posisi tertentu, penghapusan elemen berdasarkan kriteria, serta penelusuran isi list dapat dipahami dan diterapkan dengan baik. Setiap node dalam list dapat dialokasikan dan dihapus secara fleksibel tanpa perlu menggeser elemen lain seperti pada array.

Selain itu, program ini membuktikan bahwa penggunaan Single Linked List sangat berguna dalam kasus yang membutuhkan pengelolaan data secara efisien dan dinamis, seperti manajemen playlist lagu. Dengan memahami prinsip dasar linked list, mahasiswa dapat mengembangkan struktur data yang lebih kompleks seperti Doubly Linked List, Circular Linked List, Stack, atau Queue untuk kebutuhan pemrograman yang lebih luas di masa mendatang.

#### E. Referensi

FinalRoundAI. (2025). *Singly Linked List: With Code Examples and Visualization*. Diakses pada tahun 2025 dari <https://www.finalroundai.com/articles/singly-linked-list>

GeeksforGeeks. (2024). *Introduction to Linked List – Data Structure and Algorithm Tutorials*. Diakses pada tahun 2024 dari <https://www.geeksforgeeks.org/introduction-to-linked-list-data-structure-and-algorithm-tutorials/>

RumahCoding. (2024). *Linked List: Pengertian dan Implementasi Dasar*. Diakses pada tahun 2024 dari <https://rumahcoding.co.id/linked-list-pengertian-dan-implementasi-dasar/>