

Tugas Membuat Double Linked List

Rifqi Fadil Fahrial 1222646

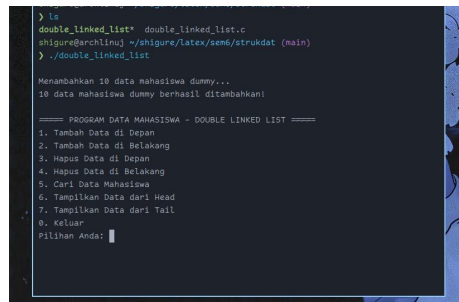
May 16, 2025

1 Soal

Buat program double link list untuk data mahasiswa dg fungsionalitas: tambah depan, tambah belakang,, hapus depan, hapus belakang, cari data, tampil data dari head, tampil data dari tail.

2 Hasil

Program double linked list ini dibuat menggunakan bahasa c yang menampilkan double link list untuk kemudahan dalam mengakses datanya



```
> ls
double_linked_list* double_linked_list.c
shigure@archlinux: ~/shigure/latex/sem6/strukdat (main)
> ./double_linked_list

Menambahkan 10 data mahasiswa dummy...
10 data mahasiswa dummy berhasil ditambahkan!

===== PROGRAM DATA MAHASISWA - DOUBLE LINKED LIST =====
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Hapus Data di Depan
4. Hapus Data di Belakang
5. Cari Data Mahasiswa
6. Tampilkan Data dari Head
7. Tampilkan Data dari Tail
8. Keluar
Pilihan Anda: █
```

Figure 1: Tampilan Utama

Aplikasi ini menginisialisasi 10 data dummy untuk mempermudah pengujian

2.1 Menambahkan Data dari depan (HEAD)

```
0: Return
Pilihan Anda: 1

Input Data Mahasiswa:
Nama: test depan
NIM: 1222646
Semester (1-14): 6
Jurusan: Depan
Fakultas: Front
Data mahasiswa berhasil ditambahkan di depan
```

Figure 2: Menambahkan Data dari depan (HEAD)

2.2 Menambahkan Data dari belakang (TAIL)

```
0: Return
Pilihan Anda: 2

Input Data Mahasiswa:
Nama: test belakang
NIM: 24356
Semester (1-14): 6
Jurusan: belakang
Fakultas: back
Data mahasiswa berhasil ditambahkan di belakang
```

Figure 3: Menambahkan Data dari belakang (TAIL)

2.3 Mencari Mahasiswa dengan NIM

===== DAFTAR MAHASISWA (DARI HEAD) =====					
ID	NAMA	NIM	SEMESTER	JURUSAN	FAKULTAS
11	test depan	1222646	6	Depan	Front
1	Ahmad Fauzi	2023001	3	Teknik Informatika	Fakultas Ilmu Komputer
2	Budi Santoso	2023002	2	Teknik Elektro	Fakultas Teknik
3	Citra Dewi	2023003	4	Sistem Informasi	Fakultas Ilmu Komputer
4	Dian Purnama	2023004	1	Manajemen	Fakultas Ekonomi
5	Eko Prasetyo	2023005	5	Akuntansi	Fakultas Ekonomi
6	Fitri Handayani	2023006	2	Ilmu Komunikasi	Fakultas Ilmu Sosial
7	Gunawan Wibisono	2023007	3	Hukum	Fakultas Hukum
8	Hani Permata	2023008	4	Kedokteran	Fakultas Kedokteran
9	Irfan Hakim	2023009	1	Arsitektur	Fakultas Teknik
10	Joko Widodo	2023010	6	Ilmu Politik	Fakultas Ilmu Sosial
12	test belakang	24356	6	belakang	back

Figure 4: Mencari Mahasiswa dengan NIM

2.4 menampilkan mahasiswa dari head

DAFTAR MAHASISWA (DARI HEAD)					
ID	NAMA	NIM	SEMESTER	JURUSAN	FAKULTAS
11	test depan	1222646	6	Depan	Front
1	Ahmad Fauzi	2023001	3	Teknik Informatika	Fakultas Ilmu Komputer
2	Budi Santoso	2023002	2	Teknik Elektro	Fakultas Teknik
3	Citra Dewi	2023003	4	Sistem Informasi	Fakultas Ilmu Komputer
4	Dian Purnama	2023004	1	Manajemen	Fakultas Ekonomi
5	Eko Prasetyo	2023005	5	Akuntansi	Fakultas Ekonomi
6	Fitri Handayani	2023006	2	Ilmu Komunikasi	Fakultas Ilmu Sosial
7	Gunawan Wibisono	2023007	3	Hukum	Fakultas Hukum
8	Hani Permata	2023008	4	Kedokteran	Fakultas Kedokteran
9	Irfan Hakim	2023009	1	Arsitektur	Fakultas Teknik
10	Joko Widodo	2023010	6	Ilmu Politik	Fakultas Ilmu Sosial
12	test belakang	24356	6	belakang	back

Figure 5: menampilkan mahasiswa dari head

2.5 menampilkan mahasiswa dari tail

DAFTAR MAHASISWA (DARI TAIL)					
ID	NAMA	NIM	SEMESTER	JURUSAN	FAKULTAS
12	test belakang	24356	6	belakang	back
10	Joko Widodo	2023010	6	Ilmu Politik	Fakultas Ilmu Sosial
9	Irfan Hakim	2023009	1	Arsitektur	Fakultas Teknik
8	Hani Permata	2023008	4	Kedokteran	Fakultas Kedokteran
7	Gunawan Wibisono	2023007	3	Hukum	Fakultas Hukum
6	Fitri Handayani	2023006	2	Ilmu Komunikasi	Fakultas Ilmu Sosial
5	Eko Prasetyo	2023005	5	Akuntansi	Fakultas Ekonomi
4	Dian Purnama	2023004	1	Manajemen	Fakultas Ekonomi
3	Citra Dewi	2023003	4	Sistem Informasi	Fakultas Ilmu Komputer
2	Budi Santoso	2023002	2	Teknik Elektro	Fakultas Teknik
1	Ahmad Fauzi	2023001	3	Teknik Informatika	Fakultas Ilmu Komputer
11	test depan	1222646	6	Depan	Front

Figure 6: menampilkan mahasiswa dari tail

2.6 menghapus mahasiswa dari head

```
Pilihan Anda: 3
Data mahasiswa dengan ID 11 berhasil dihapus dari depan
```

Figure 7: menghapus mahasiswa dari head

2.7 menghapus mahasiswa dari tail

```
Pilihan Anda: 4
Data mahasiswa dengan ID 12 berhasil dihapus dari belakang
```

Figure 8: menghapus mahasiswa dari tail

2.8 menampilkan Hasil Akhir

Pilihan Anda: 6

===== DAFTAR MAHASISWA (DARI HEAD) =====					
ID	NAMA	NIM	SEMESTER	JURUSAN	FAKULTAS
1	Ahmad Fauzi	2023001	3	Teknik Informatika	Fakultas Ilmu Komputer
2	Budi Santoso	2023002	2	Teknik Elektro	Fakultas Teknik
3	Citra Dewi	2023003	4	Sistem Informasi	Fakultas Ilmu Komputer
4	Dian Purnama	2023004	1	Manajemen	Fakultas Ekonomi
5	Eko Prasetyo	2023005	5	Akuntansi	Fakultas Ekonomi
6	Fitri Handayani	2023006	2	Ilmu Komunikasi	Fakultas Ilmu Sosial
7	Gunawan Wibisono	2023007	3	Hukum	Fakultas Hukum
8	Hani Permata	2023008	4	Kedokteran	Fakultas Kedokteran
9	Irfan Hakim	2023009	1	Arsitektur	Fakultas Teknik
10	Joko Widodo	2023010	6	Ilmu Politik	Fakultas Ilmu Sosial

Figure 9: menampilkan Hasil Akhir

3 kode program

Listing 1: Program Double Linked List

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// Struktur data untuk mahasiswa
typedef struct {
    int nomor_id;
    char nama[100];
    char nim[20];
    int semester;
    char jurusan[100];
    char fakultas[100];
} Mahasiswa;

// Struktur node untuk double linked list
typedef struct Node {
    Mahasiswa data;
    struct Node *prev;
    struct Node *next;
} Node;

// Fungsi untuk membuat node baru
Node* buatNode(Mahasiswa data) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    if (newNode == NULL) {
        printf("Alokasi memori gagal\n");
        exit(1);
    }

    newNode->data = data;
    newNode->prev = NULL;
    newNode->next = NULL;

    return newNode;
}
```

```

}

// Fungsi untuk menambah node di depan
void tambahDepan(Node** head, Node** tail, Mahasiswa data) {
    Node* newNode = buatNode(data);

    if (*head == NULL) {
        // Jika list kosong
        *head = newNode;
        *tail = newNode;
        return;
    }

    newNode->next = *head;
    (*head)->prev = newNode;
    *head = newNode;
}

// Fungsi untuk menambah node di belakang
void tambahBelakang(Node** head, Node** tail, Mahasiswa data) {
    Node* newNode = buatNode(data);

    if (*head == NULL) {
        // Jika list kosong
        *head = newNode;
        *tail = newNode;
        return;
    }

    (*tail)->next = newNode;
    newNode->prev = *tail;
    *tail = newNode;
}

// Fungsi untuk menghapus node di depan
int hapusDepan(Node** head, Node** tail) {
    if (*head == NULL) {
        printf("List kosong, tidak ada yang bisa dihapus\n");
        return 0;
    }

    Node* temp = *head;

    if (*head == *tail) {
        // Jika hanya ada satu node
        *head = NULL;
        *tail = NULL;
    } else {
        *head = (*head)->next;
        (*head)->prev = NULL;
    }

    printf("Data mahasiswa dengan ID %d berhasil dihapus dari depan\n", temp->data.nomor_id);
    free(temp);
    return 1;
}

```

```

// Fungsi untuk menghapus node di belakang
int hapusBelakang(Node** head, Node** tail) {
    if (*head == NULL) {
        printf("List kosong, tidak ada yang bisa dihapus\n");
        return 0;
    }

    Node* temp = *tail;

    if (*head == *tail) {
        // Jika hanya ada satu node
        *head = NULL;
        *tail = NULL;
    } else {
        *tail = (*tail)->prev;
        (*tail)->next = NULL;
    }

    printf("Data mahasiswa dengan ID %d berhasil dihapus dari belakang\n", temp->data.nomor_id);
    free(temp);
    return 1;
}

// Fungsi untuk mencari data berdasarkan NIM
Node* cariData(Node* head, char* nim) {
    Node* current = head;

    while (current != NULL) {
        if (strcmp(current->data.nim, nim) == 0) {
            return current;
        }
        current = current->next;
    }

    return NULL; // Data tidak ditemukan
}

// Fungsi untuk menampilkan data dari head
void tampilDariHead(Node* head) {
    if (head == NULL) {
        printf("List kosong\n");
        return;
    }

    printf("\n==== DAFTAR MAHASISWA (DARI HEAD) =====\n");
    printf("%-5s | %-30s | %-15s | %-8s | %-25s | %-25s\n", "ID", "NAMA", "NIM", "SEMESTER", "JURUSAN");
    printf("-----");

    Node* current = head;
    while (current != NULL) {
        printf("%-5d | %-30s | %-15s | %-8d | %-25s | %-25s\n",
            current->data.nomor_id,
            current->data.nama,
            current->data.nim,
            current->data.semester,
            current->data.jurusan,
            current->data.fakultas);
    }
}

```

```

        current = current->next;
    }
    printf("-----\n");
}

// Fungsi untuk menampilkan data dari tail
void tampilDariTail(Node* tail) {
    if (tail == NULL) {
        printf("List kosong\n");
        return;
    }

    printf("\n===== DAFTAR MAHASISWA (DARI TAIL) =====\n");
    printf("%-5s | %-30s | %-15s | %-8s | %-25s | %-25s\n", "ID", "NAMA", "NIM", "SEMESTER", "JURUSAN");
    printf("-----\n");

    Node* current = tail;
    while (current != NULL) {
        printf("%-5d | %-30s | %-15s | %-8d | %-25s | %-25s\n",
            current->data.nomor_id,
            current->data.nama,
            current->data.nim,
            current->data.semester,
            current->data.jurusan,
            current->data.fakultas);
        current = current->prev;
    }
    printf("-----\n");
}

// Fungsi untuk menambahkan 10 data mahasiswa dummy untuk pengujian
void tambahDummyData(Node** head, Node** tail, int* idCounter) {
    Mahasiswa dummy[10] = {
        {0, "Ahmad Fauzi", "2023001", 3, "Teknik Informatika", "Fakultas Ilmu Komputer"},
        {0, "Budi Santoso", "2023002", 2, "Teknik Elektro", "Fakultas Teknik"},
        {0, "Citra Dewi", "2023003", 4, "Sistem Informasi", "Fakultas Ilmu Komputer"},
        {0, "Dian Purnama", "2023004", 1, "Manajemen", "Fakultas Ekonomi"},
        {0, "Eko Prasetyo", "2023005", 5, "Akuntansi", "Fakultas Ekonomi"},
        {0, "Fitri Handayani", "2023006", 2, "Ilmu Komunikasi", "Fakultas Ilmu Sosial"},
        {0, "Gunawan Wibisono", "2023007", 3, "Hukum", "Fakultas Hukum"},
        {0, "Hani Permata", "2023008", 4, "Kedokteran", "Fakultas Kedokteran"},
        {0, "Irfan Hakim", "2023009", 1, "Arsitektur", "Fakultas Teknik"},
        {0, "Joko Widodo", "2023010", 6, "Ilmu Politik", "Fakultas Ilmu Sosial"}
    };

    printf("\nMenambahkan 10 data mahasiswa dummy...\n");

    for (int i = 0; i < 10; i++) {
        dummy[i].nomor_id = (*idCounter)++;
        tambahBelakang(head, tail, dummy[i]);
    }

    printf("10 data mahasiswa dummy berhasil ditambahkan!\n");
}

// Fungsi untuk membersihkan memori yang dialokasikan untuk list
void hapusList(Node** head, Node** tail) {

```

```

Node* current = *head;
Node* next;

while (current != NULL) {
    next = current->next;
    free(current);
    current = next;
}

*head = NULL;
*tail = NULL;
}

int main() {
Node* head = NULL;
Node* tail = NULL;

int pilihan;
int idCounter = 1;
Mahasiswa mhs;
char nimCari[20];

//inisialisasi data dummy
tambahDummyData(&head, &tail, &idCounter);

do {
    printf("\n===== PROGRAM DATA MAHASISWA - DOUBLE LINKED LIST =====\n");
    printf("1. Tambah Data di Depan\n");
    printf("2. Tambah Data di Belakang\n");
    printf("3. Hapus Data di Depan\n");
    printf("4. Hapus Data di Belakang\n");
    printf("5. Cari Data Mahasiswa\n");
    printf("6. Tampilkan Data dari Head\n");
    printf("7. Tampilkan Data dari Tail\n");
    printf("0. Keluar\n");
    printf("Pilihan Anda: ");
    scanf("%d", &pilihan);
    getchar(); // Membersihkan buffer

    switch(pilihan) {
        case 1: // Tambah Depan
            mhs.nomor_id = idCounter++;

            printf("\nInput Data Mahasiswa:\n");
            printf("Nama: ");
            fgets(mhs.nama, sizeof(mhs.nama), stdin);
            mhs.nama[strcspn(mhs.nama, "\n")] = '\0'; // Hapus newline

            printf("NIM: ");
            fgets(mhs.nim, sizeof(mhs.nim), stdin);
            mhs.nim[strcspn(mhs.nim, "\n")] = '\0'; // Hapus newline

            printf("Semester (1-14): ");
            scanf("%d", &mhs.semester);
            getchar(); // Membersihkan buffer

            printf("Jurusan: ");

```



```

fgets(mhs.jurusan, sizeof(mhs.jurusan), stdin);
mhs.jurusan[strcspn(mhs.jurusan, "\n")] = '\0'; // Hapus newline

printf("Fakultas: ");
fgets(mhs.fakultas, sizeof(mhs.fakultas), stdin);
mhs.fakultas[strcspn(mhs.fakultas, "\n")] = '\0'; // Hapus newline

tambahDepan(&head, &tail, mhs);
printf("Data mahasiswa berhasil ditambahkan di depan\n");
break;

case 2: // Tambah Belakang
    mhs.nomor_id = idCounter++;

    printf("\nInput Data Mahasiswa:\n");
    printf("Nama: ");
    fgets(mhs.nama, sizeof(mhs.nama), stdin);
    mhs.nama[strcspn(mhs.nama, "\n")] = '\0'; // Hapus newline

    printf("NIM: ");
    fgets(mhs.nim, sizeof(mhs.nim), stdin);
    mhs.nim[strcspn(mhs.nim, "\n")] = '\0'; // Hapus newline

    printf("Semester (1-14): ");
    scanf("%d", &mhs.semester);
    getchar(); // Membersihkan buffer

    printf("Jurusan: ");
    fgets(mhs.jurusan, sizeof(mhs.jurusan), stdin);
    mhs.jurusan[strcspn(mhs.jurusan, "\n")] = '\0'; // Hapus newline

    printf("Fakultas: ");
    fgets(mhs.fakultas, sizeof(mhs.fakultas), stdin);
    mhs.fakultas[strcspn(mhs.fakultas, "\n")] = '\0'; // Hapus newline

    tambahBelakang(&head, &tail, mhs);
    printf("Data mahasiswa berhasil ditambahkan di belakang\n");
    break;

case 3: // Hapus Depan
    hapusDepan(&head, &tail);
    break;

case 4: // Hapus Belakang
    hapusBelakang(&head, &tail);
    break;

case 5: // Cari Data
    printf("Masukkan NIM mahasiswa yang dicari: ");
    fgets(nimCari, sizeof(nimCari), stdin);
    nimCari[strcspn(nimCari, "\n")] = '\0'; // Hapus newline

    Node* hasil = cariData(head, nimCari);

    if (hasil != NULL) {
        printf("\n==== DATA MAHASISWA DITEMUKAN =====\n");
        printf("ID      : %d\n", hasil->data.nomor_id);
    }

```

```

        printf("Nama      : %s\n", hasil->data.nama);
        printf("NIM       : %s\n", hasil->data.nim);
        printf("Semester: %d\n", hasil->data.semester);
        printf("Jurusan  : %s\n", hasil->data.jurusan);
        printf("Fakultas: %s\n", hasil->data.fakultas);
    } else {
        printf("Data mahasiswa dengan NIM %s tidak ditemukan\n", nimCari);
    }
    break;

case 6: // Tampilkan dari Head
    tampilDariHead(head);
    break;

case 7: // Tampilkan dari Tail
    tampilDariTail(tail);
    break;
case 0: // Keluar
    printf("Terima kasih telah menggunakan program ini\n");
    break;

default:
    printf("Pilihan tidak valid. Silakan coba lagi.\n");
}
} while (pilihan != 0);

// Membersihkan memori sebelum keluar
hapusList(&head, &tail);

return 0;
}

```