Tugas Pendahuluan Modul 6

STRUKTUR DATA – Ganjil 2025/2026

Doubly Linked List

Ketentuan Tugas Pendahuluan

1. Tugas Pendahuluan dikerjakan secara Individu.
2. TP ini bersifat WAJIB, tidak mengerjakan = PENGURANGAN POIN JURNAL / TES ASESMEN.
3. Hanya MENGUMPULKAN tetapi TIDAK MENGERJAKAN = PENGURANGAN POIN JURNAL / TES ASESMEN.
4. Deadline pengumpulan TP Modul 6 adalah Senin, 20 Oktober 2025 pukul 06.00 WIB.
5. Tidak ada TOLERANSI KETERLAMBATAN. TERLAMBAT atau TIDAK MENGUMPULKAN TP maka DIANGGAP TIDAK MENGERJAKAN.
6. DILARANG PLAGIAT (PLAGIAT = E).
7. Kerjakan TP dengan jelas agar dapat dimengerti.
8. Untuk setiap soal nama fungsi atau prosedur WAJIB menyertakan NIM, contoh:

```
int namaFungsi_10301XXXXXXXX(...);
```

9. File diupload di LMS menggunakan format PDF dengan format:

```
TP_MOD_[XX]_NIM_NAMA.pdf
```

CP (WhatsApp):

- Falih (+62 857-2774-5199)
- Faried (+62 813-8921-4045)
- Arief (+62 852 1252 8394)
- Evan (+62 896-0298-0999)

SELAMAT MENGERJAKAN!

1 Pembuatan ADT DLL

Buatlah ADT Doubly Linked List (`DLL.h`) yang akan menyimpan data berupa angka bilangan bulat (integer), serta implementasikan beberapa subprogram dasar.

```
type infotype : integer
type address : pointer to elmlist

type elmlist <
    info : infotype
    next : address
    prev : address
>

type List <
    first : address
    last : address
>

procedure createList(in/out L : List)
function isEmpty(L : List) -> boolean
function allocate(x : infotype) -> address
procedure printInfo(in L : List)
```

2 Menambahkan Elemen di Awal dan Akhir DLL

Buatlah subprogram (`DLL.cpp`) yang mengizinkan pengguna menambahkan elemen ke dalam Doubly Linked List di awal dan di akhir list.

2.1 Sub Prorgam

1. Implementasikan prosedur `insertFirst` untuk menambahkan elemen di awal list.

```
procedure insertFirst(in/out L : List, in p : address)
{I.S. Terdefinisi list L dan alamat p yang sudah di alokasi.
 F.S. p menjadi elemen pertama dalam list L.}

algorithm
    // cek jika L kosong
    if ... then
        // set nilai first dan last dari L menjadi p
        ...
    else
        // set nilai next dari p menjadi first dari L
        ...
        // set nilai prev dari first dari L menjadi p
        ...
        // set nilai first dari L menjadi p
        ...
    endif
endprocedure
```

2. Implementasikan prosedur `insertLast` untuk menambahkan elemen di akhir list.

```
procedure insertLast(in/out L : List, in p : address)
{I.S. Terdefinisi list L dan alamat p yang sudah di alokasi.
 F.S. p menjadi elemen terakhir dalam list L.}

algorithm
    // cek jika L kosong
    if ... then
        // set nilai first dan last dari L menjadi p
        ...
    else
        // set nilai prev dari p menjadi last dari L
        ...
        // set nilai next dari last dari L menjadi p
        ...
        // set nilai last dari L menjadi p
        ...
    endif
endprocedure
```

2.2 Input & Output

Implementasikan penginputan elemen di awal dan akhir list Doubly Linked List sesuai dengan teks dibawah (teks bold bergaris bawah adalah input):

```
masukkan elemen pertama: 10
masukkan elemen kedua di awal: 5
masukkan elemen ketiga di akhir: 20

daftar elemen list: 5, 10, 20
```

3 Menghapus Elemen di Awal dan Akhir DLL

Buatlah subprogram (`DLL.cpp`) yang memungkinan pengguna untuk menghapus elemen pertama dan elemen terakhir dalam Doubly Linked List.

3.1 Sub Program

1. Implementasikan prosedur `deleteFirst` untuk menghapus elemen pertama.

```
procedure deleteFirst(in/out L : List, out p : address)
{I.S. Terdefinisi list L.
 F.S. Elemen pertama dihapus dan alamatnya disimpan pada p.}

algorithm
    // cek jika list L kosong
    if ... then
        // set nilai p menjadi nil (nullptr)
        ...
```

```
    // cek jika first dari L sama dengan last dari L
    else if ... then
        // set nilai p menjadi first dari L
        ...
        // set nilai first dan last dari L menjadi nil (nullptr)
        ...
    else
        // set nilai p menjadi first dari L
        ...
        // set nilai first dari L menjadi next dari first dari L
        ...
        // set nilai prev dari first dari L menjadi nil (nullptr)
        ...
        // set nilai next dari p menjadi nil (nullptr)
        ...
    endif
    endprocedure
endprocedure
```

2. Implementasikan prosedur `deleteLast` untuk menghapus elemen terakhir.

```
procedure deleteLast(in/out L : List, out p : address)
{I.S. Terdefinisi list L.
 F.S. Elemen terakhir dihapus dan alamatnya disimpan pada p.}

algorithm
    // cek jika list L kosong
    if ... then
        // set nilai p menjadi nil (nullptr)
        ...

    // cek jika first dari L sama dengan last dari L
    else if ... then
        // set nilai p menjadi first dari L
        ...
        // set nilai first dan last dari L menjadi nil (nullptr)
        ...
    else
        // set nilai p menjadi last dari L
        ...
        // set nilai last dari L menjadi prev dari last dari L
        ...
        // set nilai next dari last dari L menjadi nil (nullptr)
        ...
        // set nilai prev dari p menjadi nil (nullptr)
        ...
    endif
    endprocedure
endprocedure
```

3.2 Input & Output

Implementasikan penghapusan elemen di awal dan akhir list Doubly Linked List sesuai dengan teks dibawah (teks bold bergaris bawah adalah input):

```
masukkan elemen pertama: 10
```

```
masukkan elemen kedua di akhir: 15
masukkan elemen ketiga di akhir: 20

elemen pertama telah dihapus
elemen terakhir telah dihapus

daftar elemen list: 15
```

4 Menampilkan Elemen dari Depan ke Belakang dan Sebaliknya

Buatlah program yang memungkinkan pengguna memasukkan beberapa elemen ke dalam Doubly Linked List. Setelah elemen dimasukkan, tampilkan seluruh elemen dalam list dari depan ke belakang, kemudian dari belakang ke depan.

4.1 Sub Program

1. Implementasikan fungsi untuk menampilkan elemen dari depan ke belakang.

```
procedure printInfoFrontToBack(in L : List)
{I.S. Terdefinisi list L.
 F.S. Seluruh elemen dari L dicetak dari depan ke belakang dipisahkan
dengan koma (,).}
```

2. Implementasikan fungsi untuk menampilkan elemen dari belakang ke depan.

```
procedure printInfoBackToFront(in L : List)
{I.S. Terdefinisi list L.
 F.S. Seluruh elemen dari L dicetak dari belakang ke depan dipisahkan
dengan koma (,).}
```

4.2 Input & Output

Implementasikan penampilan elemen dari depan ke belakang pada list Doubly Linked List sesuai dengan teks dibawah (teks bold bergaris bawah adalah input):

```
masukkan elemen di akhir: 1
masukkan elemen di akhir: 2
masukkan elemen di akhir: 3
masukkan elemen di akhir: 4

daftar elemen dari depan ke belakang: 1, 2, 3, 4
daftar elemen dari belakang ke depan: 4, 3, 2, 1
```