**Capaian Pembelajaran**: Mahasiswa mampu mengimplementasikan Double Linked List.

**Pokok Bahasan**: Double Linked List

**Tujuan Tugas**: Implementasi Double Linked List dengan menggunakan notasi algoritma

**Aturan**:

1. Kerjakan secara individu
2. Jawaban ditulis di kertas kemudian difoto dan diupload di LMS

**Jumlah Soal**: 1 Soal

**SOAL 1**

Untuk menyimpan data akun yang registrasi ke aplikasi X digunakan representasi Linear Double Linked List. Data akun yang disimpan terdiri dari username dan password. Username harus UNIK. Berikut ilustrasinya:

Last

First

xyz

001pqr

userAbc

123abc

user1

password

1. Buat definisi struktur datanya (ADT)!
2. Buat implementasi fungsi/ prosedur berikut untuk melakukan registrasi/ sign up akun:
3. function createNewElm (akun: infotype) 🡪 address

*{Membuat (mengalokasikan) sebuah elemen baru. Jika alokasi berhasil, info dari elemen berisi akun, prev dan next dari elemen berisi nil. Kembalikan (return) address dari elemen baru tsb.}*

1. procedure InsertLast (In p: address; In/Out L: List)

*{Menambahkan sebuah elemen baru beralamat p sebagai elemen terakhir list L.*

*IS: List tidak kosong, p sudah dialokasi, p≠nil, next(p)=nil.*

*FS: p adalah elemen terakhir list L.}*

1. function findAkun (username: String, L: List) 🡪 address

*{Mencari akun dengan username tertentu.*

*IS: Terdefinisi List L dan username yang akan dicari. List mungkin kosong.*

*FS: Return alamat dimana akun ditemukan, return nil jika tidak ditemukan. }*

1. procedure signUp (In akun: infotype, In/Out L: List)

*{IS: Terdefinisi List L serta data akun baru. List mungkin kosong.*

*FS: Jika akun belum terdaftar, sisipkan sebagai elemen terakhir List L; jika sudah, tampilkan pesan “Account has been registered.” }*

1. Buat implementasi prosedur berikut untuk men-remove sebuah akun:
2. procedure deleteFirst (In/Out L: List, Out p: address)

*{ IS: Terdefinisi List L yang memiliki lebih dari 1 elemen.*

*FS: p adalah elemen pertama yang dihapus dari List L. }*

1. procedure deleteAfter (In q: address, Out p: address)

*{ IS: Terdefinisi List L yang memiliki lebih dari 2 elemen., q ≠ nil, prev(q) ≠ nil, next(q) ≠ last.*

*FS: p adalah elemen setelah q yang dihapus dari List L. }*

1. procedure deleteLast (In/Out L: List, Out p: address)

*{ IS: Terdefinisi List L yang memiliki lebih dari 1 elemen.*

*FS: p adalah elemen terakhir yang dihapus dari List L. }*

1. procedure removeAkun (In username: String, In/Out L: List

*{ IS: List mungkin kosong. Jika tidak kosong, minimal memiliki 1 elemen.*

*FS: Akun dengan username tsb di-dealokasi dari List L.*

*Notes:*

* *Gunakan function b(iii) untuk mencari akun.*
* *Akun yang dihapus mungkin merupakan elemen pertama, tengah, atau terakhir.}*