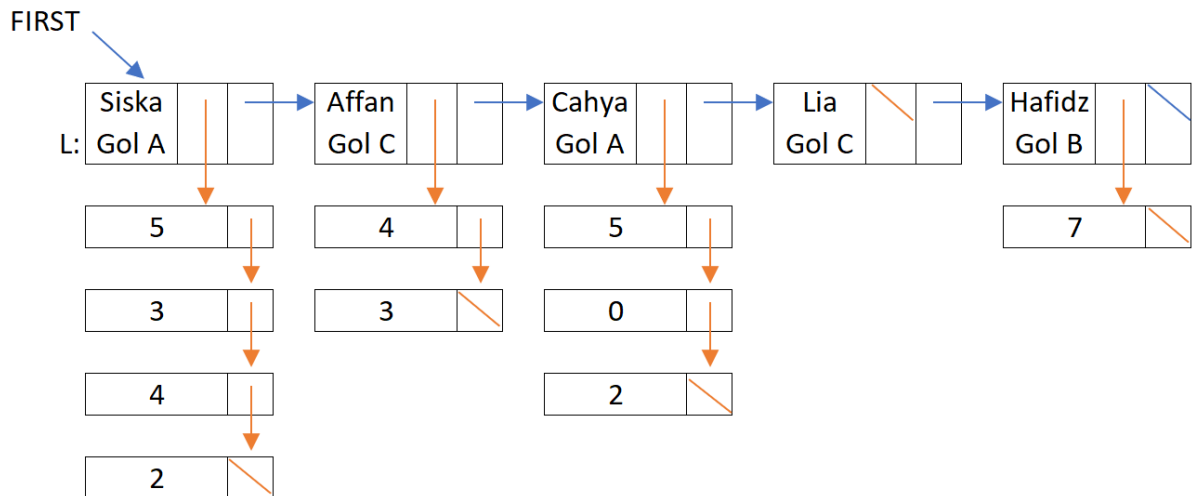


MK STRUKTUR DATA
TUGAS MULTI LINKED LIST



Terdapat data jumlah penjualan mobil masing-masing sales nya perbulan di sebuah showroom mobil. Data tersebut dimasukkan kedalam multi linked list 1-N

Tugas Anda adalah membuat implementasi fungsi procedure di bawah ini

Type adr_jual: < pointer to elm_jual >

Type elm_jual : < info : integer,
Next : adr_jual >

Type sales : < nama, gol : string >

Type adr_sales : < pointer to elm_sales >

Type elm_sales : < info : sales,
Next : adr_sales
nextJual : adr_jual >

Type mll : < First : adr_sales >

List_Sales : mll

Asumsi FUNgsi dan procedure dibawah ini sudah terdefinisi (Anda tinggal menggunakan)

1. Function Search_Sales (List_Sales : mll, In nama_Sales : string) → adr_sales
{Fungsi ini akan mencari nama sales pada list sales, jika ditemukan maka akan dikembalikan alamatnya, jika tidak ditemukan maka akan dikembalikan NIL}
2. Procedure insert_last_Sales (In/out List_Sales : mll, in S : adr_Sales)
{I.S. Terdefinisi list sales yang mungkin kosong, dan elemen sales baru yang disimpan oleh pointer S, yang akan diinsertkan menjadi elemen sales paling akhir pada list sales
F.S. elemen baru menjadi elemen sales paling akhir pada list sales}

Lengkapilah implementasi dari fungsi procedure berikut ini :

Procedure Create_list (In/Out List_Sales: mll)

{I.S. –

F.S. Dihasilkan sebuah multi linked list 1 ke N dengan pointer first yang NIL}

Kamus Data

Algoritma

// NIL kan pointer First dari list

.....

Procedure new_elm_sales (In info : sales, In/Out S : adr_sales)

{I.S. terdefinisi data sales dan pointer yang akan menyimpan alamat elemen dari data sales yang baru

F.S. data sales baru sudah menjadi elemen dan alamatnya disimpan oleh pointer S}

Kamus Data

Algoritma

//alokasi elemen sales yang alamatnya disimpan oleh pointer S

.....

//NIL kan pointer next dari elemen

.....

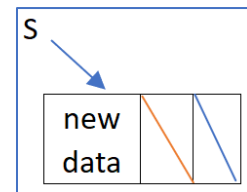
//NIL Kan pointer next_jual dari elemen

.....

//masukkan data sales ke dalam info elemen

.....

.....



Procedure new_elm_jual (In info : integer, In/Out J : adr_jual)

{I.S. terdefinisi data jumlah penjualan dan pointer yang akan menyimpan alamat elemen dari data penjualan yang baru

F.S. data sales baru sudah menjadi elemen dan alamatnya disimpan oleh pointer S}

Kamus Data

Algoritma

//alokasi elemen jual yang alamatnya disimpan oleh pointer J

.....

//NIL kan pointer next dari elemen

.....

//masukkan data penjualan ke dalam info elemen

.....



Procedure Insert_new_Penjualan (In/Out List_Sales : mll, In S : adr_Sales, In J : adr_jual)

{I.S. Terdefinisi list sales yang tidak kosong, dan elemen penjualan baru yang akan disisipkan menjadi data penjualan pada sales S. Data penjualan pada sales S tidak kosong.

F.S. Elemen penjualan baru akan disisipkan menjadi data penjualan **paling akhir** di sales pada elemen S. CLUE : **Konsep Insert Last**}

ILUSTRATION



Kamus Data

.....

Algoritma

// cari lokasi elemen penjualan paling akhir

.....

.....

.....

.....

//sambungkan elemen paling akhir ke elemen yang baru

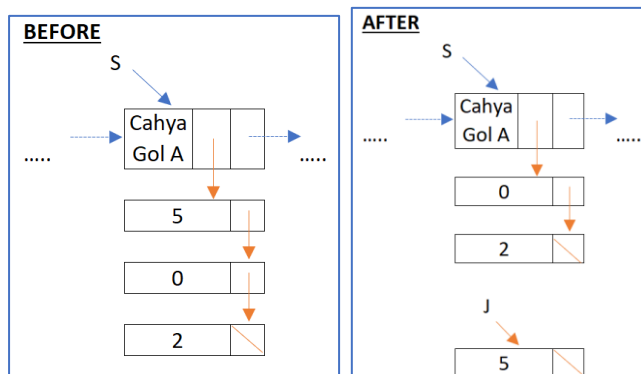
.....

Procedure Delete_First_Penjualan (In/Out List_Sales : mll, In S: adr_sales, out J : adr_jual)

{I.S. Terdefinisi list sales yang tidak kosong, dan elemen sales yang akan dihapus elemen penjualannya yang berada di posisi paling pertama

F.S. Elemen penjualan pertama dari sales S dihapus dan alamatnya disimpan di pointer J.

CLUE : **Konsep delete first** }



Kamus

Algoritma

.....

Procedure Show_data_sales (In list_sales : mll)

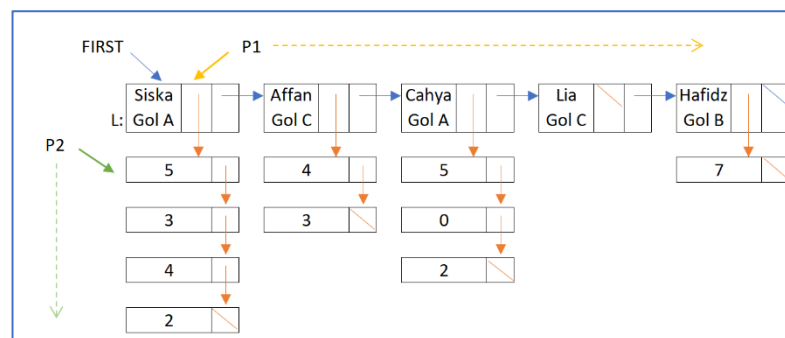
{I.S. Terdefinisi list sales yang tidak kosong}

F.S. Data penjualan setiap sales tampil ke layar}

Tampilan :

```
Siska
5 3 4 2
Affan
4 3
Cahya
5 0 2
Lia
Hafidz
7
```

HINT : Anda membutuhkan 2 pointer, pointer yang akan menelusuri elemen sales satu persatu, dan setiap kali pointer itu mengunjungi sebuah elemen sales, ada pointer kedua yang akan menelusuri elemen data penjualan dari sales tsb. Setelah selesai penelusuran elemen penjualan di sebuah elemen sales, maka pointer pertama tadi bergeser ke sales selanjutnya, dan mengulangi Langkah yang sama, hingga semua sales terkunjungi.



Kamus Data

.....

Algoritma

// setting awal pointer sales

.....

//loop pointer sales menelusuri semua elemen sales

.....

//mengoutputkan info sales

.....

//setting start pointer data penjualan di elemen penjualan pertama dari elemen sales saat ini

.....

//Loop pointer data penjualan menelusuri semua data penjualan dari sales saat ini

.....

//mengoutputkan info penjualan

.....

//pindah ke data penjualan berikutnya

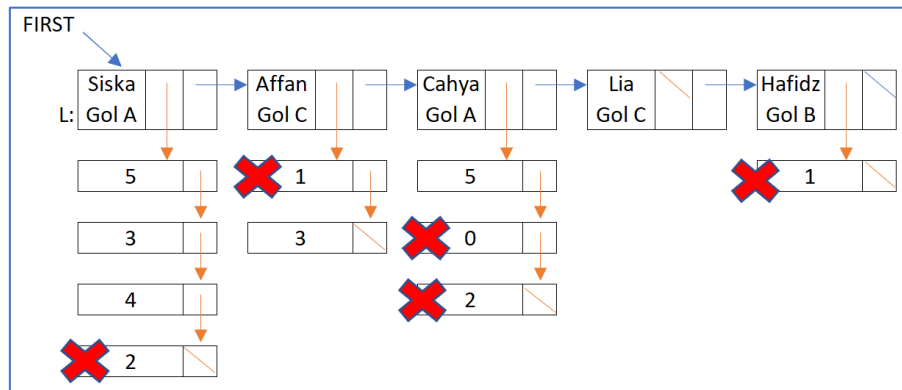
.....
//pindah ke data sales berikutnya

.....

Procedure Delete_Penjualan (In /out list_sales : mll)

{I.S. Terdefinisi list sales yang tidak kosong}

F.S. Data penjualan kurang dari 3 dihapus dari elemen penjualan}



Clue : telusuri semua data penjualan dari setiap sales, jika menemukan data penjualan kurang dari 3 maka lakukan penghapusan dengan cara memanggai procedure delete sesuai kondisi elemennya (Apakah elemen nya di paling awal, atau di paling akhir, atau diantaranya)

Kamus Data

//fungsi procedure ini sudah terdefinisi, Anda tinggal panggil saja sesuai kebutuhan

Procedure Delete_First_Penjualan (In/Out List_Sales : mll, In S: adr_sales, out J : adr_jual)

{I.S. Terdefinisi list sales yang tidak kosong, dan elemen sales yang akan dihapus elemen penjualannya yang berada di posisi paling pertama}

F.S. Elemen penjualan pertama dari sales S dihapus dan alamatnya disimpan di pointer J. CLUE : Konsep delete first }

Procedure Delete_Last_Penjualan (In/Out List_Sales : mll, In S: adr_sales, out J : adr_jual)

{I.S. Terdefinisi list sales yang tidak kosong, dan elemen sales yang akan dihapus elemen penjualannya yang berada di posisi paling akhir}

F.S. Elemen penjualan paling akhir dari sales S dihapus dan alamatnya disimpan di pointer J. CLUE : Konsep delete last }

Procedure Delete_after_Penjualan (In/Out List_Sales : mll, In S: adr_sales, In prec: adr_jual, out J : adr_jual)

{I.S. Terdefinisi list sales yang tidak kosong, pointer prec}

F.S. Elemen penjualan yang berada setelah pointer prec dihapus. CLUE : Konsep delete after }

```

// setting awal pointer sales
.....
//loop pointer sales menelusuri semua elemen sales
.....

//setting start pointer data penjualan di elemen penjualan pertama dari elemen
sales saat ini
.....

//Loop pointer data penjualan menelusuri semua data penjualan dari seles saat ini
.....

//Pengecekan apakah data penjualan saat ini kurang dari 3 atau tidak.
.....
//Action ketika kurang dari 3 adalah mengecek apakah elemen nya
berada di data paling pertama di sales tersebut, jika ya, lakukan
delete first, dst
IF (.....) THEN
.....
ELSE IF (.....) THEN
.....
ELSE
.....

//berpindah ke data penjualan berikutnya
.....

//berpindah ke data sales berikutnya
.....

```