List Rekursif dalam Konteks Prosedural (2)

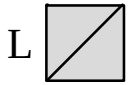
IF2110/IF2111 – Algoritma dan Struktur Data
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

List sebagai Struktur Data Rekursif

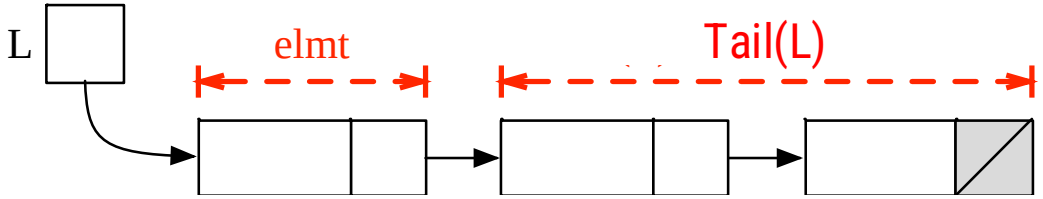
Definisi rekursif list linier:

- **Basis**: list kosong adalah list
- **Rekurens**: list tidak kosong terdiri atas sebuah elemen dan sisanya adalah list

List L kosong

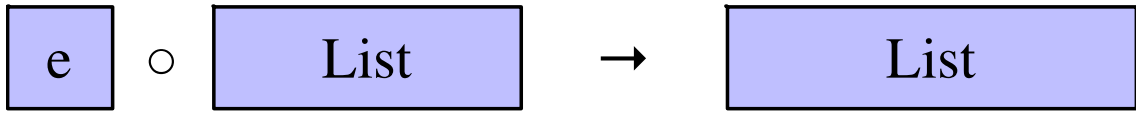


List L dengan tiga elemen



Struktur Data List

type List: [ ] atau [e ○ List]



Primitif dasar (ingat kembali list dalam pemrograman fungsional):

- Selektor: head, tail (≠ head, tail pada queue)
- Konstruktor: kons○, kons●
- Primitif-primitif lain: copy, concat, dll.

Selektor

```
function head(l: List) → ElType
{ Mengirimkan elemen pertama sebuah list l yang tidak kosong }
KAMUS LOKAL
ALGORITMA
    → l↑.info
```

```
function tail(l: List) → List
{ Mengirimkan list l tanpa elemen pertamanya, mungkin yang dikirimkan adalah sebuah list kosong }
KAMUS LOKAL
ALGORITMA
    → l↑.next
```

Konstruktor – Kons○

```
function konso(e: ElType, l: List) → List
{ Mengirimkan list l dengan tambahan e sebagai elemen pertamanya }
{ Jika alokasi gagal, mengirimkan l }
```

```
KAMUS LOKAL
    p: Address
```

```
ALGORITMA
    p ← newNode(e)
    if (p = NIL) then
        → l
    else
        { Insert First }
        p↑.next ← l
        → p
```

Konstruktor – Kons●

```
function kons●(l: List, e: ElType) → List
{ Mengirimkan list l dengan tambahan e sebagai elemen terakhir }
{ Jika alokasi gagal, mengirimkan l }
```

```
KAMUS LOKAL

```

```
ALGORITMA
    if isEmpty(l) then { insert ke list kosong }
        → newNode(e)
    else
        l↑.next ← kons●(tail(l), e)
        → l
```

Konstruktor – Kons●

```
procedure kons●(input/output l: List, input e: ElType)
{ Mengirimkan list l dengan tambahan e sebagai elemen terakhir }
{ Jika alokasi gagal, mengirimkan l }
```

```
KAMUS LOKAL

```

```
ALGORITMA
    if isEmpty(l) then { insert ke list kosong }
        l ← alokasi(e)
    else
        kons●(l↑.next, e)
```

Primitif Lain: Copy – 1 (versi fungsi)

```
function copy(l: List) → List
{ Mengirimkan salinan list l }
{ Jika alokasi gagal, mengirimkan l }
KAMUS LOKAL

ALGORITMA
    if (isEmpty(l)) then { Basis 0 }
        → NIL
    else { Rekurens }
        → konso(head(l), copy(tail(l))
```

Primitif Lain: Copy – 2 (versi procedure)

```
procedure mCopy(input lin: List, output lout: List)
{ I.S. lin terdefinisi }
{ F.S. lout berisi salinan dari lin }
{ Proses: menyalin lin ke lout }
{ Jika alokasi gagal, Lout adalah ??? }
```

```
KAMUS LOKAL
    lTemp: List
```

```
ALGORITMA
    if (isEmpty(lin)) then { Basis – 0 }
        lout ← NIL
    else { Rekurens }
        mCopy(tail(lin), lTemp)
        lout ← konso(head(lin), lTemp)
```

Primitif Lain: Concat – 1 (versi fungsi)

```
function concat(l1, l2: List) → List
{ Mengirimkan salinan hasil konkatenasi list l1 dan l2 }
```

```
KAMUS LOKAL

```

```
ALGORITMA
    if (isEmpty(l1)) then { Basis }
        → copy(l2)
    else { Rekurens }
        → konso(head(l1), concat(tail(l1), l2))
```

Primitif Lain: Concat – 2 (versi procedure)

```
procedure mConcat (input l1, l2: List, output result: List)
{ I.S. l1, l2 terdefinisi }
{ F.S. result adalah hasil melakukan konkatenasi l1 dan l2 dengan cara "disalin" }
{ Proses: Menghasilkan salinan hasil konkatenasi list l1 dan l2 }
```

```
KAMUS LOKAL
    lTemp: List
```

```
ALGORITMA
    if (isEmpty(l1)) then  { Basis – 0 }
        result ← copy(l2)
    else { Rekurens }
        mConcat(tail(l1), l2, lTemp)
        result ← konso(head(l1), lTemp)
```

Potongan header file (1)

```
#ifndef LISTREC_H
#define LISTREC_H

#include "boolean.h"
#include <stdio.h>

#define NIL NULL

typedef int ElType;
typedef struct node* Address;
typedef struct node {
    ElType info;
    Address next;
} Node;

typedef Address List;

/* Selektor */
#define INFO(p) (p)->info
#define NEXT(p) (p)->next
```

Potongan header file (2)

```
/* Manajemen Memori */

Address newNode(ElType x);
/* Mengirimkan address hasil alokasi sebuah elemen */
/* Jika alokasi berhasil, maka address tidak NIL, dan misalnya
   menghasilkan p, maka INFO(p)=x, NEXT(p)=NIL */
/* Jika alokasi gagal, mengirimkan NIL */
```

Potongan header file (3)

```
/* Pemeriksaan Kondisi List */
boolean isEmpty(List l);
/* Mengirimkan true jika l kosong dan false jika l tidak kosong */
int isOneElmt(List l);
/* Mengirimkan true jika l berisi 1 elemen dan false jika > 1 elemen atau kosong */

/* Primitif-Primitif Pemrosesan List */
ElType head(List l);
/* Mengirimkan elemen pertama sebuah list l yang tidak kosong */
List tail(List l);
/* Mengirimkan list l tanpa elemen pertamanya, mungkin mengirimkan list kosong */
List konso(List l, ElType e);
/* Mengirimkan list l dengan tambahan e sebagai elemen pertamanya. e dialokasi terlebih dahulu.
   Jika alokasi gagal, mengirimkan l */

/* Fungsi dan Prosedur Lain */
...

#endif
```