

# List Rekursif dalam Konteks Prosedural (1)

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

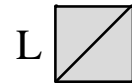
# List sebagai Struktur Data Rekursif

Definisi rekursif list linier:

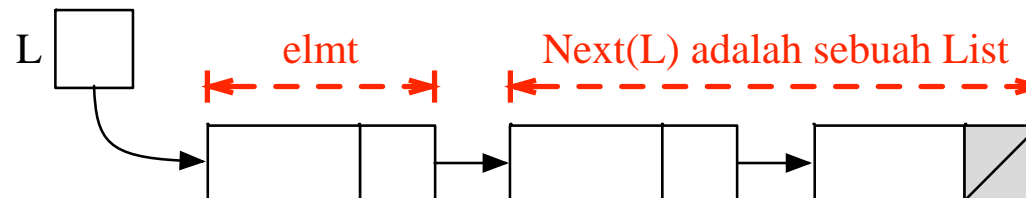
**Basis:** list kosong adalah list

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

List L kosong



List L dengan tiga elemen



# Struktur Data List untuk Pemrosesan secara Rekursif (Notasi Algoritmik)

## KAMUS

*{ List direpresentasi dg pointer }*

type ElType: ... { terdefinisi }

type Address: ... { terdefinisi }

type Node: < info: ElType,  
                    next: Address >

type List: Address

*{ Deklarasi nama untuk variabel kerja }*

l: List

p: Address               *{ address untuk traversal }*

*{ Maka First(L) adalah l*

*Next(p), Info(p) tergantung representasi fisik yang digunakan }*

# Struktur Data List untuk Pemrosesan secara Rekursif (Bahasa C, pointer)

```
#define NIL NULL

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

typedef int ElType;
typedef struct node* Address;
typedef struct node {
    ElType info;
    Address next;
} Node;
/* Definisi list: */
/* List kosong: l = NIL */

typedef Address List;
```

# Primitif Dasar: Pemeriksaan List Kosong

## Notasi Algoritmik (rep. berkait)

**function** isEmpty(l: List) → **boolean**  
{ *Tes apakah sebuah list l kosong.*  
  *Mengirimkan true jika list kosong,*  
  *false jika tidak kosong }*

**KAMUS LOKAL**

-

**ALGORITMA**

→ (l = NIL)

## Bahasa C (rep. berkait dgn. pointer)

```
boolean isEmpty(List l) {  
  /* Tes apakah sebuah list l kosong.  
   Mengirimkan true jika list kosong,  
   false jika tidak kosong */  
  /* Kamus Lokal */  
  
  /* Algoritma */  
  return (l == NIL);  
}
```

# Studi Kasus-1: displayList

## Notasi Algoritmik

```
procedure displayList(input l: List)
{ I.S. l terdefinisi }
{ F.S. Setiap elemen list diprint }
KAMUS LOKAL
ALGORITMA
    if (isEmpty(l)) then { Basis 0 }
        { tidak melakukan apa-apa }
    else { Rekurens }
        output(l↑.info)
        displayList(l↑.next)
```

# Studi Kasus-2a: NbElmtList (1)

## Versi fungsi

function length(l: List) → integer  
{ Mengirimkan banyaknya elemen list l, Nol jika l kosong }

KAMUS LOKAL

ALGORITMA

```
if (isEmpty(l)) then { Basis 0 }  
    → 0  
else { Rekurens }  
    → 1 + length(l↑.next)
```

# Studi Kasus-2b: NbElmtList (2)

Versi **prosedur**, dengan hasil diletakkan pada parameter output

```
procedure length(input l: List, output n: integer)
```

```
{ I.S. l terdefinisi
```

```
  F.S. n berisi banyaknya elemen list }
```

KAMUS LOKAL

```
  n1: integer
```

ALGORITMA

```
  if (isEmpty(l)) then { Basis 0 }
```

```
    n ← 0
```

```
  else { Rekurens }
```

```
    length(l↑.next, n1)
```

```
    n ← 1 + n1
```



# Studi Kasus-2c: NbElmtList (3) - 1

Versi prosedur, dengan akumulator

```
procedure lengthAcc(input l: List, input acc: integer, output n: integer)  
{ I.S. l terdefinisi  
  F.S. n berisi banyaknya elemen list }
```

KAMUS LOKAL

ALGORITMA

```
  if (isEmpty(l)) then { Basis 0 }  
    n ← acc  
  else { Rekurens: Next element, Proses }  
    length(l↑.next, acc+1, n)
```

# Studi Kasus-2c: NbElmtList (3) - 2

## Pemanggilan **lengthAcc**

procedure length(input l: List, output n: integer)

{ *I.S. l terdefinisi*

*F.S. n berisi banyaknya elemen list l*

*Proses: Memanfaatkan lengthAcc }*

**KAMUS LOKAL**

**ALGORITMA**

lengthAcc(l, 0, n)

# Studi Kasus-3: search

function search(l: List, x: ElType) → boolean  
{ Mengirim true jika x adalah anggota list, false jika tidak }

**KAMUS LOKAL**

**ALGORITMA**

if (isEmpty(l)) then { Basis 0 }  
    → false  
else  
    → (l↑.info = x) or search(l↑.next, x)

# Studi Kasus-3: search, versi 2

function search(l: List, x: ElType) → boolean  
{ Mengirim true jika x adalah anggota list, false jika tidak }

KAMUS LOKAL

ALGORITMA

```
if (isEmpty(l)) then { Basis 0 }  
    → false  
else  
    if (l↑.info = x) then { Basis 1 }  
        → true  
    else { Rekurens }  
        → search(l↑.next, x)
```

# Studi Kasus-4: Delete Elemen

Prosedur menghapus semua elemen list bernilai X

**procedure** deleteX(**input/output** l: List, **input** x: infotype)

{ I.S. l dan x terdefinisi }

{ F.S. semua elemen l yang bernilai x dihapus dari l }

**KAMUS LOKAL**

p: Address

**ALGORITMA**

**if** (isEmpty(L)) **then** { Basis 0 }

{do nothing}

**else** { Rekurens }

deleteX(l↑.next, x)

**if** l↑.info=x **then**

p ← l

l ← l↑.next

dealokasi(p)