

LIST OF LIST

A horizontal bar spanning the width of the slide, composed of five colored segments: dark blue, medium blue, red, green, and dark blue.

LIST OF LIST

List of list adalah list yang:

- Mungkin kosong
 - $S = []$
- Mungkin terdiri dari sebuah elemen yang disebut atom dan sisanya adalah list of list
 - $S = [A, B, [C, D, E], F]$
- Mungkin terdiri dari sebuah elemen berupa list dan sisanya adalah list of list
 - $S = [[A, B, C], D, E, [F, G]]$

Contoh list:

- ▣ List kosong : []
- ▣ List dengan elemen 3 atom : [ad,a,b]
- ▣ List dengan elemen **list kosong**, **L1**, **L2**, dan **sebuah atom**

[[],[a,b,c],[d,e],f]

DEFINISI DAN SPESIFIKASI LIST OF LIST



IsEmpty

- ▣ IsEmpty : List of List \rightarrow boolean
 - ▣ IsEmpty(S) : Benar jika S adalah List of list kosong
- ▣ $X = [] \rightarrow$ IsEmpty(X) bernilai benar
- ▣ $Y = [a, b, [c, d]] \rightarrow$ IsEmpty(Y) bernilai salah

```
def is_empty_LoL(S):  
    return S == []
```

```
def is_empty_LoL(S):
```

```
    return S==[]
```

```
L1=[]
```

```
L2=[]]
```

```
L3=[['s',2,3],2,[5,'b']]
```

```
print(is_empty_LoL(L1)) → True
```

```
print(is_empty_LoL(L2)) → False
```

```
print(is_empty_LoL(L3)) → False
```

IsAtom

- IsAtom : list of list \rightarrow boolean
 - IsAtom(S) menghasilkan true jika list adalah atom
- $X=[a] \rightarrow \text{IsAtom}(X)$ bernilai benar
- $Y=[]$ atau $Y=[a,b]$ atau $Y=[a,[b,c]]$
 - IsAtom(Y) bernilai Salah

```
def is_atom(S):
```


```
    return not(is_empty_LoL(S)) and jml_elemen_list(S)==1
```

```
def is_atom(S):  
    return not(is_empty_LoL(S)) and jml_element_list(S)==1  
  
L1=['a']  
L2=['b',5]  
L3=[['s',2,3],2,[5,'b']]  
print(is_atom(L1)) → True  
print(is_atom(L2)) → False  
print(is_atom(L3)) → False
```


Islist

- `IsList(S)` : list of list \rightarrow boolean
 - `IsList(S)` : menghasilkan true jika S adalah sebuah list (bukan atom)
- `X=[a,b,c]` , `X=[a,[b,c]]` \rightarrow `IsList(X)` bernilai benar
- `Y=[a]` , `Y=[2]` \rightarrow `IsList(Y)` bernilai salah

```
def is_List(S):  
    return not(is_atom(S))
```



```
L1=['a']
```

```
L2=['b',5]
```

```
L3=[['s',2,3],2,[5,'b']]
```

```
print(is_List(L1)) → False
```

```
print(is_List(L2)) → True
```

```
print(is_List(L3)) → True
```

KonsLo

- ▣ KonsLo : List, List of List \rightarrow List of List
- ▣ KonsLo(L,S) : membentuk list baru dengan list (L) yang diberikan sebagai elemen pertama list of list (S)
- ▣ $L=[1,2,3]$, $S=[3,5,[3,8]]$
 - ▣ $S'=\text{KonsLo}(L,S)=[[1,2,3],3,5,[3,8]]$

```
def konso_LoL(L,S):  
    if is_empty_LoL(S):  
        return [L]  
    else:  
        return [L]+S
```

KonsLo

- ▣ $\text{KonsL} \bullet$: List of List, List \rightarrow List of List
- ▣ $\text{KonsL} \bullet(S, L)$: membentuk list baru dengan list (L) yang diberikan sebagai elemen terakhir list of list (S)
- ▣ $L=[1,2,3]$, $S=[3,5,[3,8]]$
 - ▣ $S'=\text{KonsL} \bullet(S, L)=[3,5,[3,8],[1,2,3]]$

```
def konsi_LoL(L,S):  
    if is_empty_LoL(S):  
        return [L]  
    else:  
        return S+[L]
```

FirstList

- FirstList : List of List tidak kosong \rightarrow List
- FirstList(S) : menghasilkan elemen pertama list (mungkin sebuah list atau atom)
- $X=[a,[b,c,e]] \rightarrow \text{FirstList}(X): [a]$
- $X=[[a,b],e,[d,c]] \rightarrow \text{firstList}(X) : [a,b]$

```
def first_List(S):  
    if not(is_empty_LoL(S)):  
        return S[0]
```

```
L3=[['s',2,3],2,['5','b']]  
print(first_List(L3)) → ['s', 2, 3]
```

TailList

- ▣ TailList : list of list tidak kosong \rightarrow list of list
- ▣ TailList(S) : sisa list of list S tanpa elemen pertama lis S
- ▣ $X=[a,[b,c,e]] \rightarrow \text{TailList}(X): [b,c,e]$
- ▣ $X=[[a,b],e,[d,c]] \rightarrow \text{TailList}(X) : [e,[d,c]]$

```
def tail_List(S):  
    if not(is_empty_LoL(S)):  
        return S[1:]
```

LastList

- ▣ LastList : list of list tidak kosong \rightarrow list of list
- ▣ LastList(S) : elemen terakhir dari list of list S, mungkin list atau atom
- ▣ $X=[a,[b,c,e]] \rightarrow \text{LastList}(X): [b,c,e]$
- ▣ $X=[[a,b],e] \rightarrow \text{LastList}(X) : [e]$

```
def last_List(S):  
    if not(is_empty_LoL(S)):  
        return S[-1:]
```


HeadList

- ▣ HeadList : list of list tidak kosong \rightarrow list of list
- ▣ HeadList(S) : menghasilkan sisa list of list tanpa elemen terakhir list
- ▣ $X=[a,[b,c,e]] \rightarrow \text{HeadList}(X): [a]$
- ▣ $X=[[a,b],e,[d,c]] \rightarrow \text{HeadList}(X) : [[a,b],e]$

```
def head_List(S):  
    if not(is_empty_LoL(S)):  
        return S[:-1]
```

Contoh 1

Mencek kesamaan dua buah list of list

KESAMAAN

IsEqS (L1,S2)

DEFINISI PREDIKAT

IsEqS : 2 List of list \rightarrow boolean

{IsEqS (S1,S2) true jika S1 identik dengan S2 : semua elemennya sama }

{ Basis : kedua list kosong : \rightarrow true

salah satu list kosong : \rightarrow false

Rekurens :

$S1 \quad \boxed{L1} \circ \boxed{\text{Tail}(S1)}$

$S2 \quad \boxed{L2} \circ \boxed{\text{Tail}(S2)}$

L1 dan L2 adalah atom : $L1=L2$ and $\text{IsEqS}(\text{TailList}(S1),\text{TailList}(S2))$

L1 dan L2 adalah list : $\text{IsEqS}(L1,L2)$ and $\text{IsEqS}(\text{TailList}(S1),\text{TailList}(S2))$

Else:false

}

Contoh 1

Mencek kesamaan dua buah list of list

Realisasi

IsEqs(S1,S2) :

depend on S1,S2

IsEmpty(S1) and IsEmpty(S2) : true

not IsEmpty(S1) and IsEmpty(S2) : false

IsEmpty(S1) and not IsEmpty(S2) : false

Not IsEmpty(S1) and not IsEmpty(S2)

depend on FirstList(S1),Firstlist(S2)

IsAtom(FirstList(S1)) and IsAtom(FirstList(S2))

FirstList(S1)=FirstList(S2) and IsEqS(TailList(S1),TailList(S2))

IsList(FirtsList(S1)) and IsList(FirtsList(S2))

IsEqS(FirstList(S1),FirtsList(S2)) and IsEqS(TailList(S1), TailList(S2))

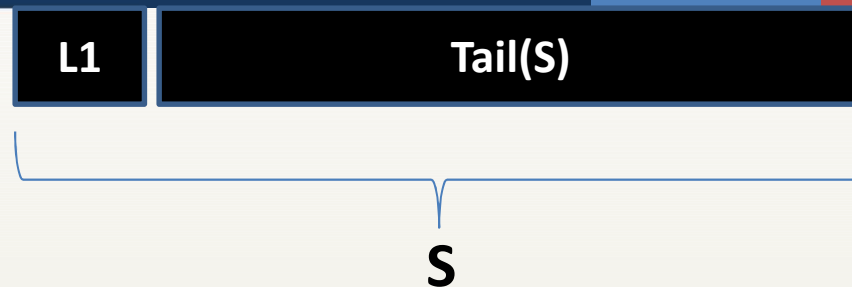
Else

False

Contoh 2

- Mencek keanggotaan sebuah elemen terhadap list of list
- Contoh:
 - Apakah elemen **c** didalam $s = []$?
 - Apakah elemen **c** didalam $s = [c, b, [a, d, e]]$?
 - Apakah elemen **c** didalam $s = [a, b, [c, d, e]]$?

Contoh 2



- Jika S Kosong \rightarrow false
- Jika L1 adalah atom dan $L1=A \rightarrow$ true
- Jika L1 bukan atom \rightarrow A anggota L1 atau A anggota dari Tail(S)

Contoh 2

KEANGGOTAAN

IsMemberS (A,S)

DEFINISI PREDIKAT

IsMemberS : elemen, List of list \rightarrow boolean

{ IsMemberS (A,S) true jika A adalah anggota S }

{ Basis : list kosong : \rightarrow false

Rekurens :

$\boxed{L1} \circ \boxed{\text{Tail}(S)}$

L1 adalah atom dan $A = L1 : \text{true}$

L1 bukan atom : A anggota L1 or IsMemberS(A, TailList(S))

}

REALISASI

IsMemberS (A, S) :

depend on S

IsEmpty(S) : false

Not IsEmpty(S) :

depend on FirstList(S)

IsAtom(FirstList(S)) : A = FirstList(S)

IsList(FirstList(S)) : IsMember(A, FirstList(S))

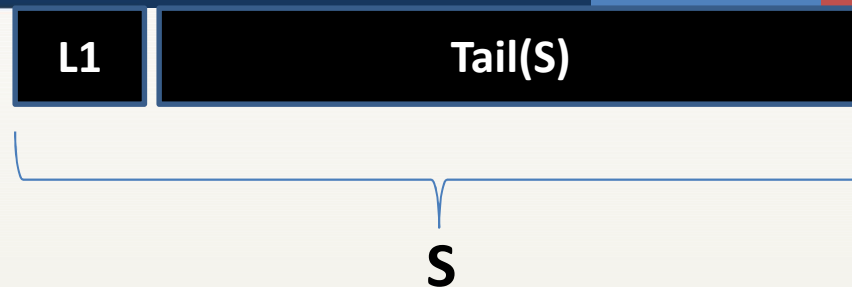
or IsMemberS(A, TailList(S))

{ dengan IsMember(A,L) adalah fungsi yang mengirimkan true jika A adalah elemen list L }

Contoh 3

- Mencek keanggotaan sebuah list (L) terhadap list of list (S)

Contoh 3



- ▣ Jika L dan S list kosong \rightarrow true
- ▣ L tidak kosong dan S kosong \rightarrow false
- ▣ L kosong dan S tidak Kosong \rightarrow false
- ▣ L1 atom \rightarrow IsMemberLS(Taillist(L,S))
- ▣ L1 bukan atom
 - ▣ L1=L \rightarrow true
 - ▣ L1 \neq L \rightarrow IsMemberLS(L,TailList(S))

Contoh 3

KEANGGOTAAN	IsMemberLS (L,S)
DEFINISI PREDIKAT IsMemberLS : List, <u>List of list</u> \rightarrow <u>boolean</u> $\{ \text{IsMemberLS } (L,S) \text{ true jika } L \text{ adalah anggota } S \}$ $\{ \text{Basis} : L \text{ dan } S \text{ list kosong} : \rightarrow \underline{\text{true}}$ $\quad \quad \quad L \text{ atau } S \text{ tidak kosong} : \underline{\text{false}}$ Rekurens : <div style="border: 1px solid black; padding: 5px; display: inline-block;"> $L1 \circ \text{Tail}(S)$ </div> $L1 \text{ adalah atom} : \text{IsMemberLS}(L, \text{TailList}(S))$ $L1 \text{ bukan atom} : L1=L : \text{true}$ $\quad \quad \quad L1 \neq L : \text{IsMemberLS}(L, \text{TailList}(S))$ $\}$	
REALISASI IsMemberLS (L, S) : <u>depend on S</u> $\text{IsEmpty}(L) \text{ and } \text{IsEmpty}(S) : \underline{\text{true}}$ $\text{not IsEmpty}(L) \text{ and } \text{IsEmpty}(S) : \underline{\text{false}}$ $\text{IsEmpty}(L) \text{ and } \text{not IsEmpty}(S) : \underline{\text{false}}$ $\text{not IsEmpty}(L) \text{ and } \text{not IsEmpty}(S) :$ $\quad \text{if } (\text{IsATOM}(\text{FirstList}(S))) \text{ then } \text{IsMemberLS}(\text{Taillist}(L, S))$ $\quad \text{else } \{ \text{IsLIST}(\text{FirstList}(S)) \}$ $\quad \quad \text{If } \text{IsEqual}(L, \text{FirstList}(S)) \text{ then } \underline{\text{true}}$ $\quad \quad \text{else } \text{IsMemberLS}(L, \text{Taillist}(S))$ { dengan IsEqual(L1,L2) adalah fungsi yang mengirimkan true jika list L1 sama dengan list L }	

Contoh 4

Menghapus sebuah elemen (atom) dari list of list

- Menghapus elemen a pada list of list S
 - Jika $S = []$ maka $S = []$
 - Jika $L1$ atom dan $L1 = a$ maka $\text{TailList}(S)$ tanpa a
 - Jika $L1$ atom dan $L1$ tidak sama dengan a maka $L1$
 - $(\text{TailList}(S)$ tanpa a)
 - Jika $L1$ adalah list maka $(L1$ tanpa a) ○ $(\text{TailList}(S)$ tanpa a)

Contoh 4

Menghapus sebuah elemen (atom) dari list of list

HAPUS*ELEMEN

Rember*(a,S)

DEFINISI

Rember*: elemen, List of list \rightarrow List of list

{ Rember (a,S) menghapus sebuah elemen bernilai a dari semua list S }*

{ List kosong tetap menjadi list kosong }

{ Basis : list kosong : \rightarrow () }

Rekurens :

a

S $\boxed{L1} o \boxed{Tail(S)}$

L1 adalah atom : $L1 = a : TailList(S)$ tanpa a

$L1 \neq a : L1 o (TailList(S) \text{ tanpa } a)$

L1 adalah list : $(L1 \text{ tanpa } a) o (TailList(S) \text{ tanpa } a)$

}

REALISASI

Rember*(a,L) :

if IsEmpty(S) then S

else if IsList(FirstList(S))

then KonsLo(Rember*(a,FirstList(S)), Rember*(a,TailList(S)))

else { elemen pertama S adalah atom }

if FirstElmt(S) = a then

Rember*(a,TailList(S))

else

KonsLo (FirstElmt(S),Rember*(a,Tail(S))

Contoh 5 :

Menentukan nilai maksimum dari list of list

- Jika S mempunyai 1 elemen:
 - Jika mempunyai 1 atom maka `firstlist(S)`
 - Jika mempunyai 1 list maka `max(firstlist(S))`
- Jika mempunyai beberapa elemen
 - Jika elemen pertama atom maka `max2(firstList(S), Max(TailList(S)))`
 - Jika elemen pertama adalah list maka `max2(Max(firstList(S)), Max(TailList(S)))`

Contoh 5 :

Menentukan nilai maksimum dari list of list

ELEMEN BERNILAI MAKSIMUM

Max(S)

DEFINISI

Max List of list tidak kosong \rightarrow integer

{ Max (S) menghasilkan nilai elemen (atom) yang maksimum dari S }

{ Basis : list dengan satu elemen E1

E1 adalah atom : nilai E1

E1 adalah list : Max(E1)

Rekurens :

a

S *L1* o *Tail(S)*

L1 adalah atom : Max2(L1,Max(Tail(S)))

L1 adalah list : Max2 (Max(L1), Max(Tail(S)))

}

{Fungsi antara }

Max2 2 integer \rightarrow integer

{Max2(a,b) menghasilkan nilai maksimum a dan b }

Contoh 5 :

Menentukan nilai maksimum dari list of list

REALISASI

Max2(a,b) :

If a>=b then

a

Else

b

Max(S) :

if IsOneElmt(S) then {Basis 1 }

if IsAtom(FirstList(S)) then

FirstList(S)

Else { List }

Max(FirstList(S))

Else {Rekurens }

if IsAtom(FirstList(S)) then {First elemen adalah atom }

Max2(FirstList(S),Max(TailList(S))

Else { Firs element adalah List }

Max2(Max(FirstList(S)), Max(TailList(S))