List

List tidak kosong

List of [tipe data]

List of integer >= 0

List of character

List of real

Dst

Selektor:

1. FirstElmt
2. LastElmt
3. Tail
4. Head

FirstElmt → Tail

LastElmt → Head

[9, 7, 5, 3, 10, 12]

[10, 12]

FilterGenap(L):

```
if IsEmpty(L) then
    []
else
    if FirstElmt(L) mod 2 = 0 then
        Konso(FirstElmt(L), FilterGenap(Tail(L))
    else
        FilterGenap(Tail(L))
```

Atom

List

[8, 7, [12, 3, 5], 6, 9, [13, 15, 11]]

[8, [12], 6, []] → [8, 12, 6]

```
FilterGenapLoL(S):
    if IsEmpty(S) then
        []
    else
        if isAtom(FirstList(S)) then
            if FirstList(S) mod 2 = 0 then
                KonsLo(FirstList(S), FilterGenapLoL(TailList(S)))
            else
                FilterGenapLoL(TailList(S))
        elif isList(FirstList(S)) then
            KonsLo(FilterGenap(FirstList(S)), FilterGenapLoL(TailList(S)))
```

Pohon N-aire

Prefix → Awal:

(7(2(8), 5(10, 9), 3(11, 13), 15))

atau

(7(2(8()), 5(10(), 9()), 3(11(), 13()), 15()))

Postfix → Setelah:

(((8)2, (11, 13)3, (10, 9)5, 15)7)

Infix → tengah tengah:

((()2(8), (10)5(9)) 7 ((11)3(13), ()15()))

Tree → <akar: elemen, anak: list of Tree>

[7, [[2, [[8, []]]], [5, [[10, []], [9, []]]], [3, [[11, []], [13, []]]], [15, []]]]

IMNChild: Pohon N-aire(P), elemen(X) → boolean

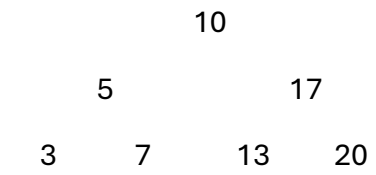
```
if isTreeEmpty(P) then
    False
else
    IsMemberNode(FirstElmt(P), X) or IMNChild(Tail(P), X)
```

IsMemberNode: Pohon N-aire (P), elemen(X) → boolean

Misal X = 15

```
if isEmpty(P) then
    false
else
    if Akar(P) = X then
        true
    else
        IMNChild(Anak(P))
```

Pohon Biner



Prefix:

(10 (5(3)(7)) (17(13)(20)))

Infix:

(((3)5(7))10((13)17(20)))

Postfix:

(((3)(7)5)((13)(20)17)10)

BST:

Anak dari Akar sebelah Kiri < Akar

sebelah Kanan > Akar

lambda

fungsi kecil

```
Max2(a,  b):
        (a + b + abs(a-b)) / 2
```

λx,y.(x + y + abs(x - y)) / 2

```
LIST = [7, 8, 10, 2, 16, 5, 4, 9]
FilterList(L, F):
        if isEmpty(L) then
                []
        else
                if F(FirstElmt(L)) then
                        Konso(FirstElmt(L), FilterList(Tail(L), F))
                else
                        FilterList(Tail(L), F)
FilterList(LIST, λx.x mod  2 = 0)
FilterList(LIST, λx.x > 5)
```