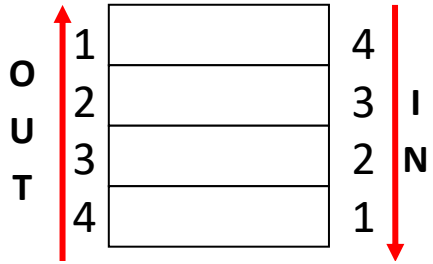
PROGRAM STUDI
TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

Mata Kuliah
Algoritma dan
Struktur Data

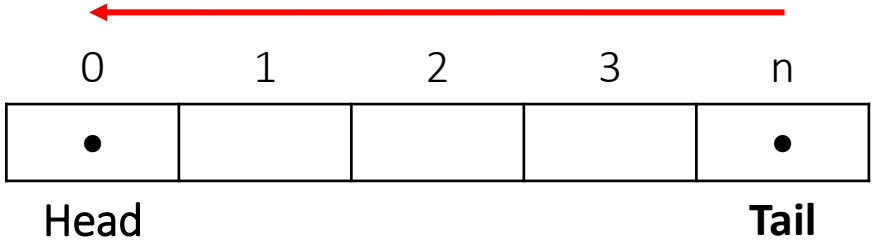
TREE

Tim Pengampu Mata Kuliah Algoritma dan Struktur Data

Struktur Data Linier

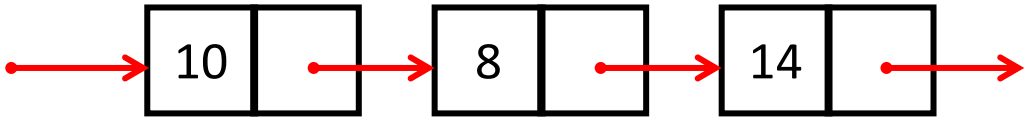


STACK



QUEUE

ARRAY



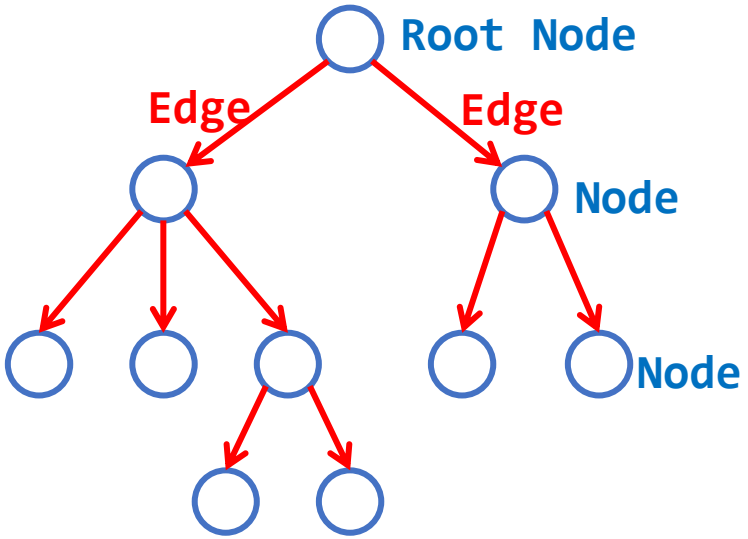
LINKED LIST

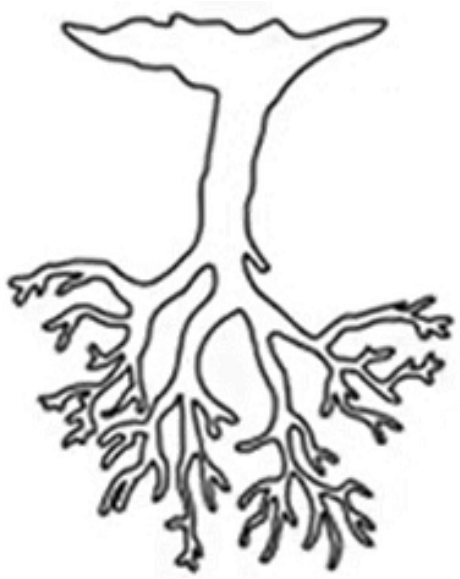
Tree

- Pohon adalah **struktur data hirarki**
- Tree adalah struktur data yang terdiri dari entitas disebut node yang terkait melalui sebuah edge
- Node paling atas disebut dengan root

Tree

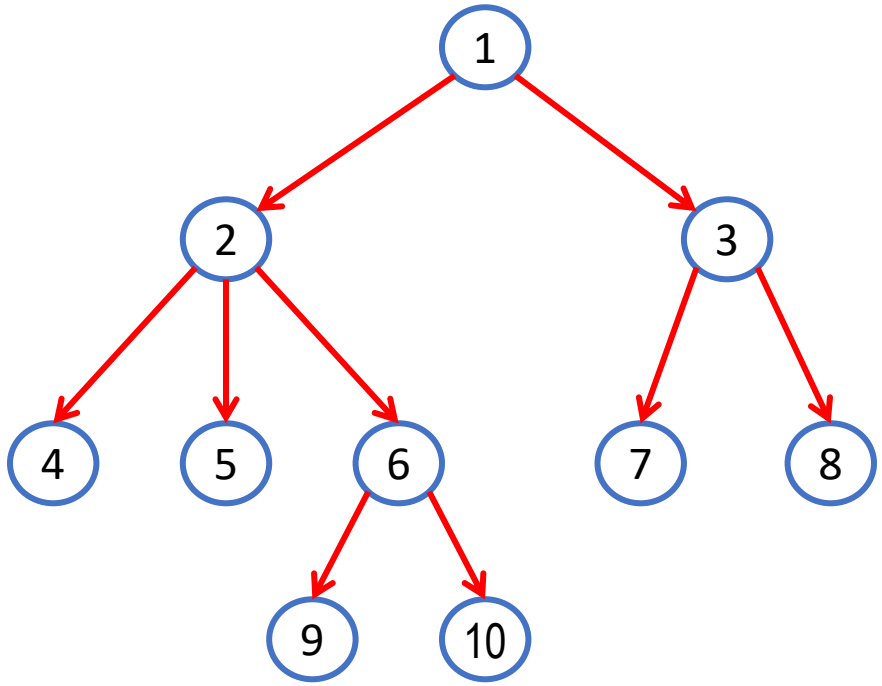
- Pohon adalah **struktur data hirarki**
- Tree adalah struktur data yang terdiri dari entitas disebut node yang terkait melalui sebuah edge
- Node paling atas disebut dengan root





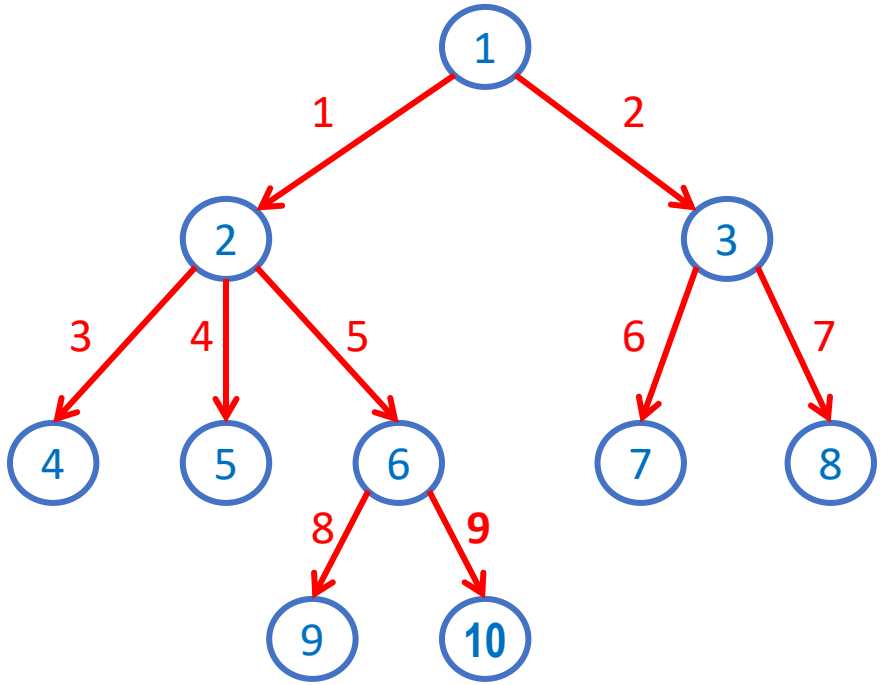
Tree

- Node dengan posisi yang lebih tinggi disebut dengan parent dan yang lebih rendah (dibawah parent) disebut dengan children
- Node dengan posisi yang sama disebut sibling
- Node dengan posisi paling rendah dari node lainnya disebut leaf



- 1 adalah root
- 1 adalah parent dari 2 dan 3
- 2 dan 3 adalah children dari 1
- 2 adalah parent dari 4, 5, dan 6
- 4, 5, dan 6 adalah sibling
- 7 dan 8 adalah children dari 3
- 7 dan 8 adalah sibling
- 9 dan 10 adalah leaf

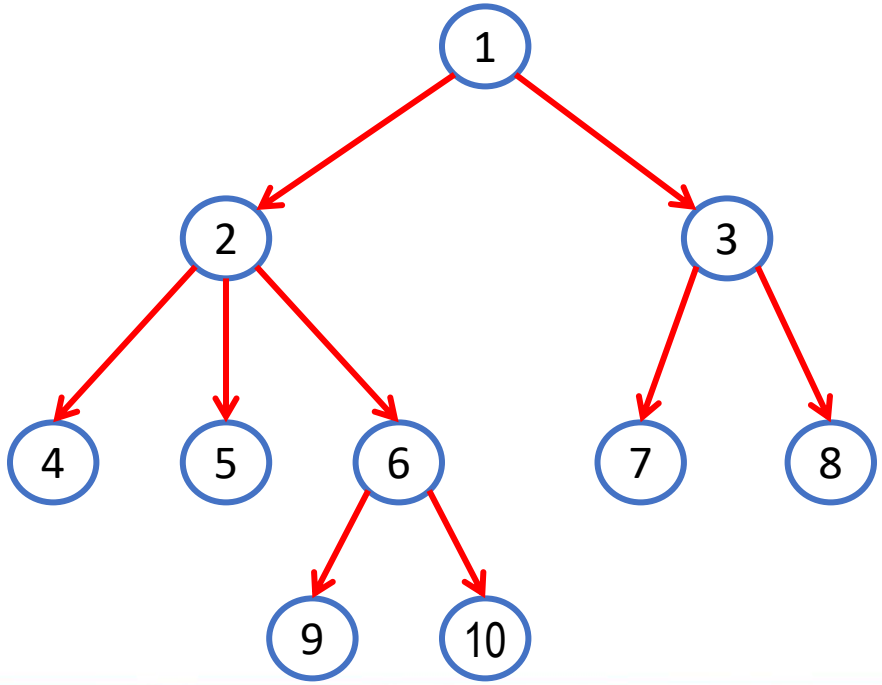
Tree



- Tree mempunyai:
  - N node
  - N – 1 edge
- Jumlah node adalah 10
- Jumlah edge adalah 9

Tree

- Depth of Node: jumlah edge dari root ke node
- Height of Node: jumlah edge terpanjang dari node ke leaf
- Height of tree: height of root node



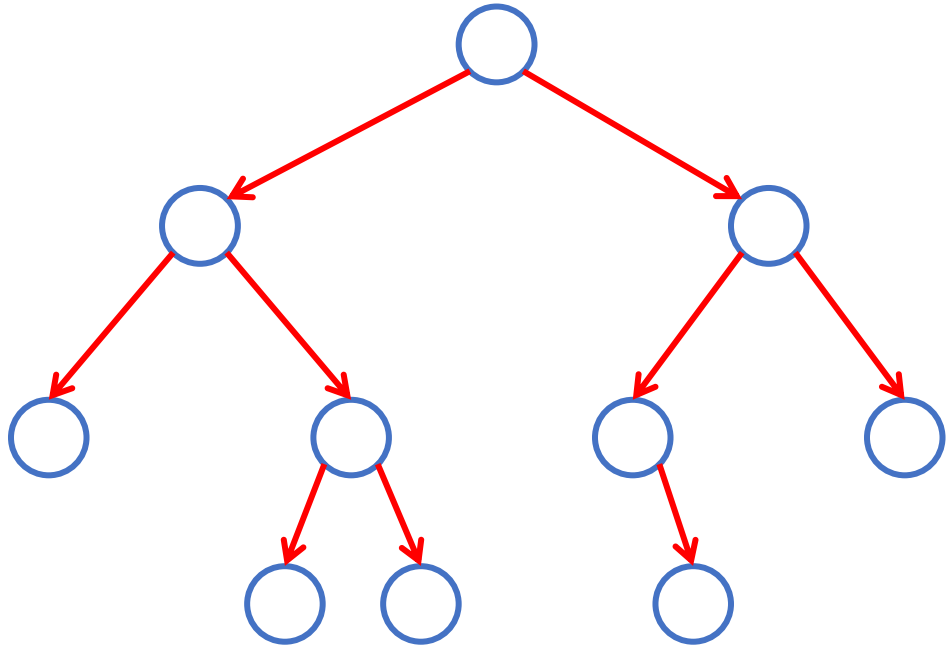
- Depth of node 1 adalah 0
- Height of node 1 adalah 3

- Depth of node 6 adalah 2
- Height of node 6 adalah 1

- Depth of node 9 adalah 3
- Height of node 9 adalah 0
- Height of tree adalah 3

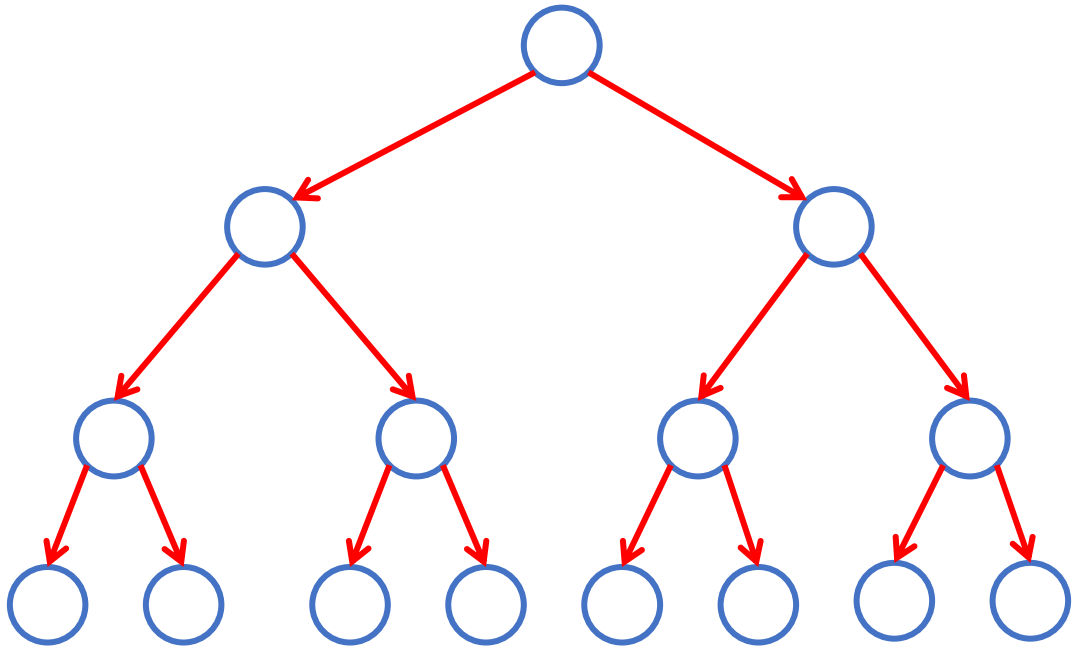
Binary Tree

- Binary tree adalah tree dimana setiap node mempunya paling banyak **2 children**
- Children dari setiap node disebut **left-child** dan **right-child**



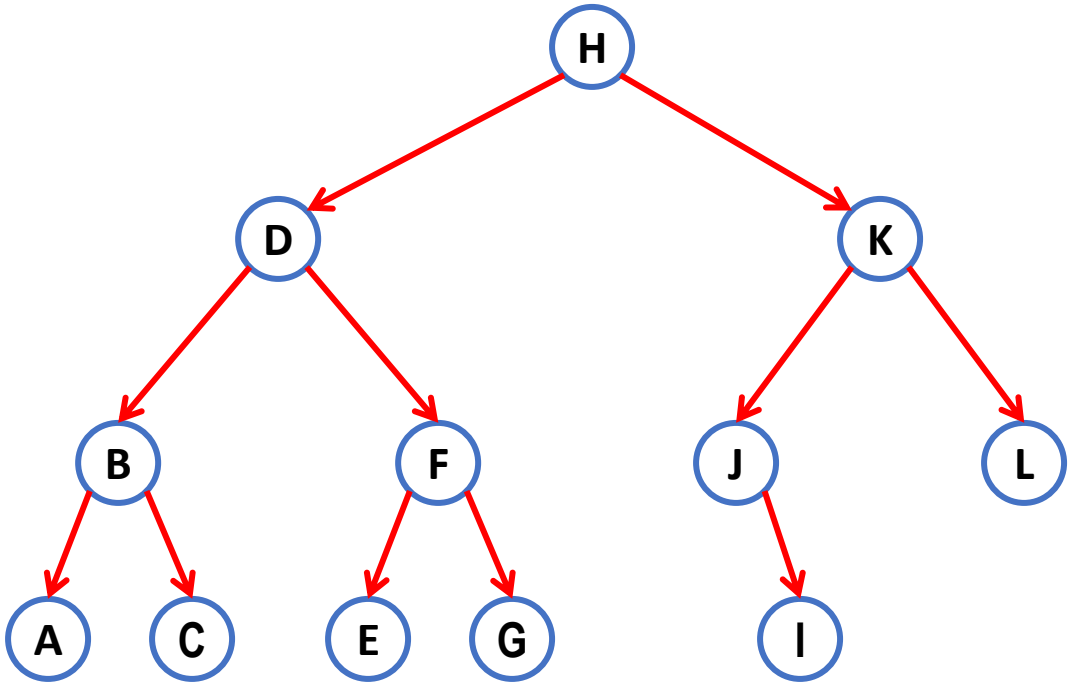
Perfect Binary Tree

- Semua level pada tree terisi lengkap



Binary Tree Traversal

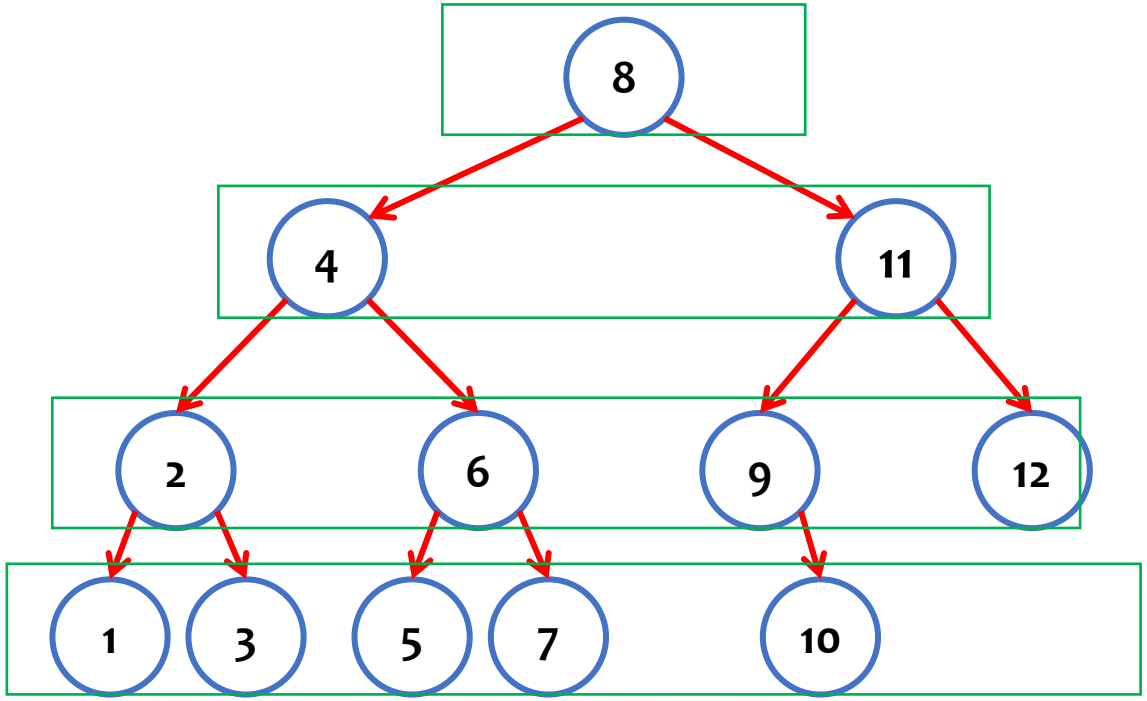
- Breadth First: Level Order
- Depth First:
  - Pre order
  - In order
  - Post order



Binary Tree Traversal

- Level Order Traversal

  Mengunjungi setiap node dari level teratas kemudian bergerak ke node **sebelah kiri**, kemudian node sebelah kanan pada level dibawahnya



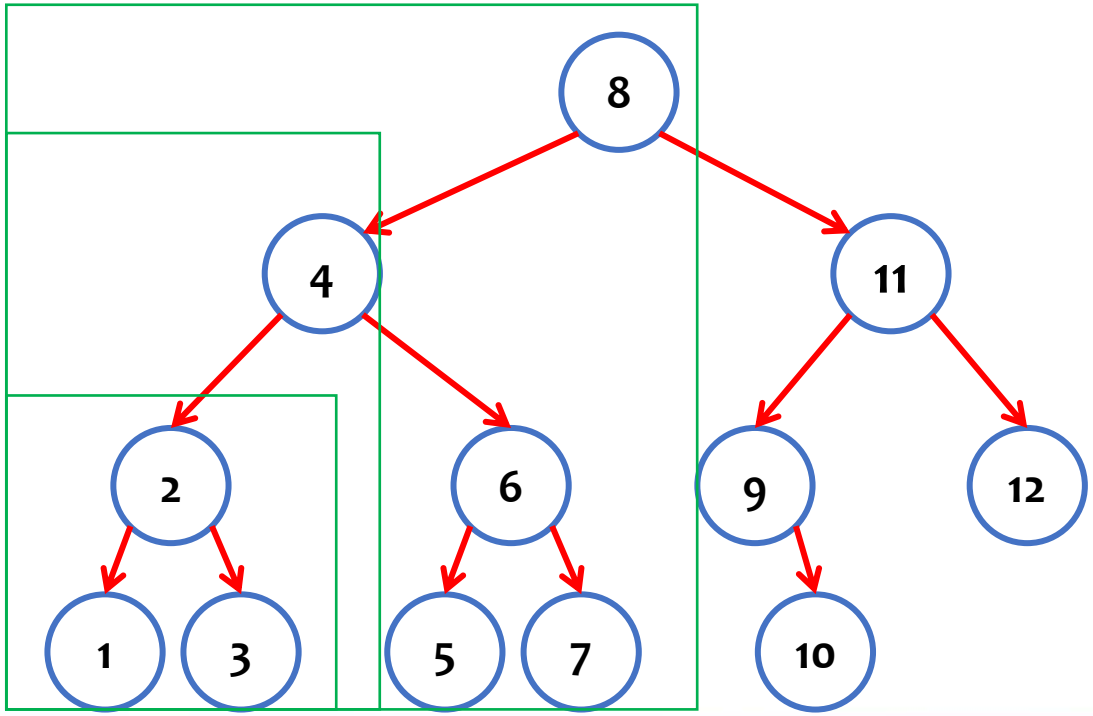
8 – 4 – 11 – 2 – 6 – 9 – 12 – 1 – 3 – 5 – 7 – 10

Binary Tree Traversal

- Pre Order Traversal

  Mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:

  Data/Parent/Root → Left Children → Right Children



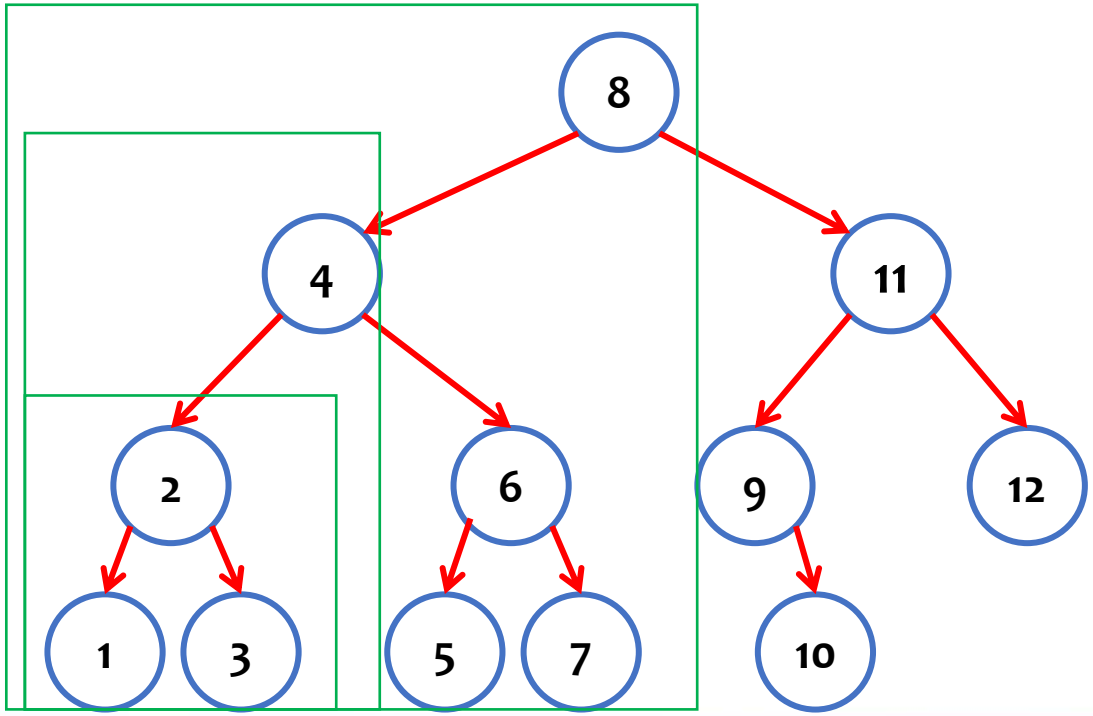
8 – 4 – 2 – 1 – 3 – 6 – 5 – 7 – 11 – 9 – 10 – 12

Binary Tree Traversal

- In Order Traversal

  Mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:

  Left Children → Data/Parent → Right Children



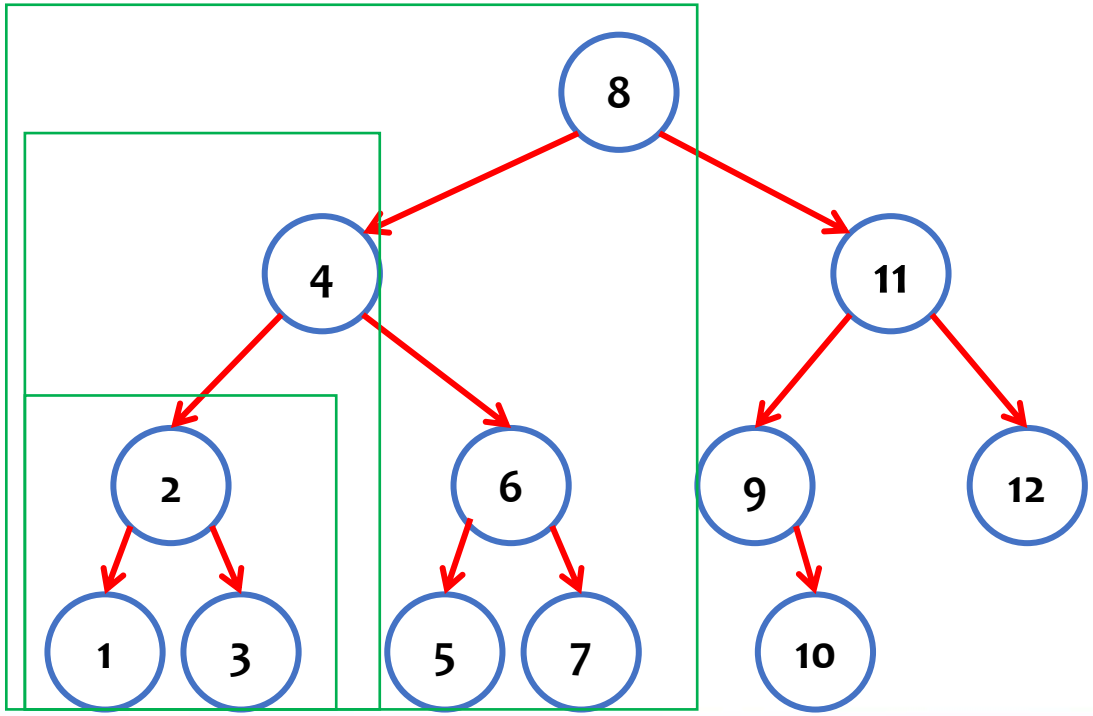
1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12

Binary Tree Traversal

- In Order Traversal

  Mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:

  Left Children → Data/Parent → Right Children



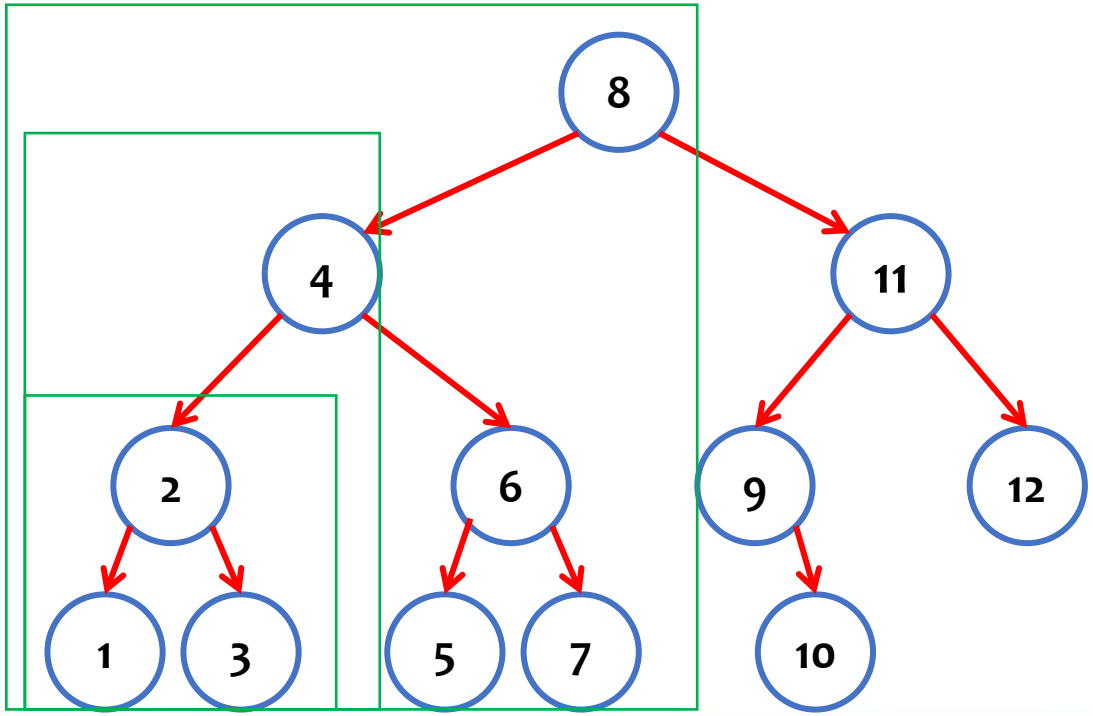
1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12

Binary Tree Traversal

- Post Order Traversal

  Mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:

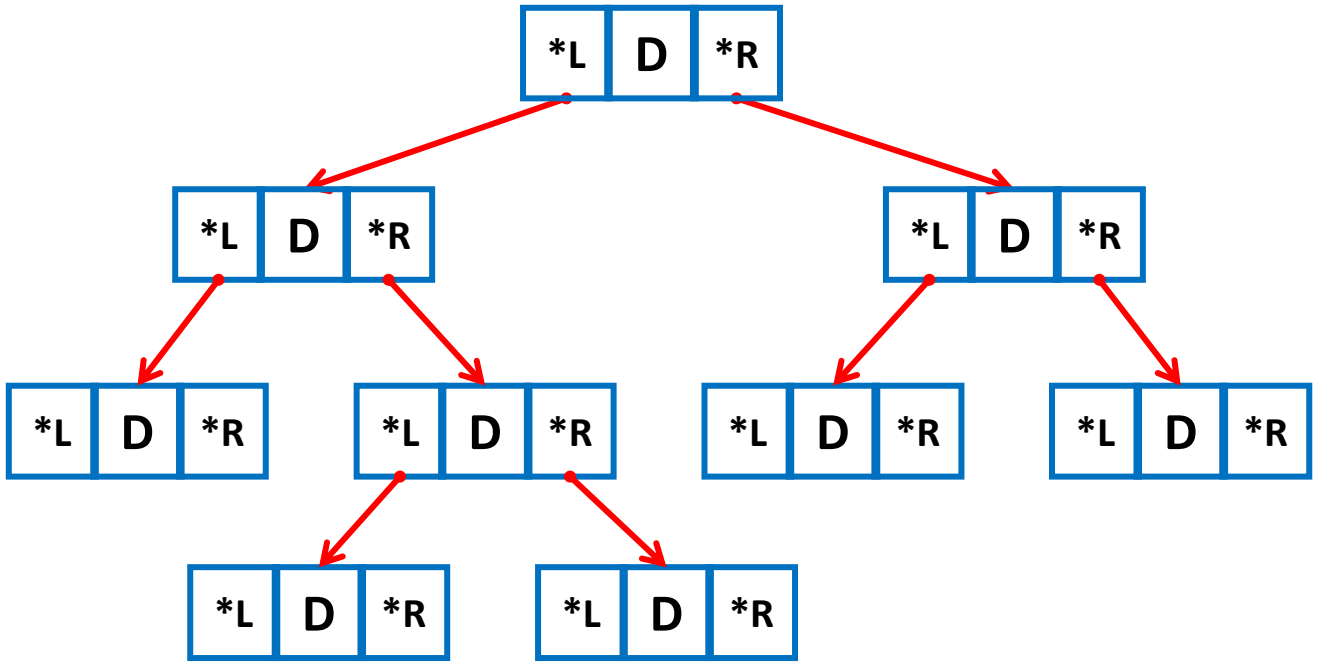
  Left Children → Right Children → Data/Parent



1 – 3 – 2 – 5 – 7 – 6 – 4 – 10 – 9 – 12 – 11 – 8

Pembentukan Binary Tree

- Binary tree dibentuk dengan node yang mempunyai **D**ata dan dua buah pointer/link (***L**eft dan ***R**ight)



Pembentukan Binary Tree

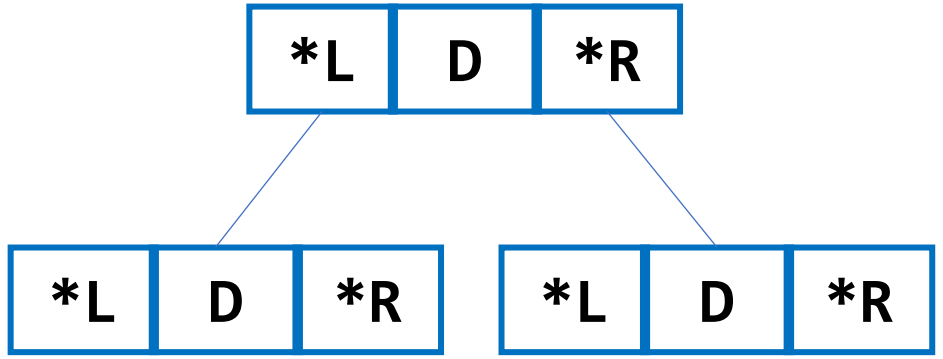
1. Deklarasi node terdiri dari : Left, Data, Right
2. Buat rangkaian atau node Binary Tree terdiri dari:
   - Node baru
   - Insert element atau root
   - Print PreOrder
   - Print InOrder
   - Print PostOrder

Deklarasi Node


```
{ADT tree}
{definisi tipe tree, left=null, right=null}
type Node:<data: integer,
          left: address,
          right: address>
```

Node Baru Binary Tree

```
{definisi binary tree}
{buat node baru}
type BinaryTree:
procedure NodeBaru(input: data)
{buat node baru terdiri data,left,right}
        return Node(input:data)
procedure insert(root: address,input:data)
{memasukkan elemen ke node}
  if root == null then:
        return NodeBaru()
  else:
        if data < root.data then:
                root.left = insert(data,root.left)
        else:
                root.right = insert(data,root.right)
        return root
```



Node Baru Binary Tree

```
{definisi binary tree}
{buat node baru}
type BinaryTree:
procedure NodeBaru(input: data)
{buat node baru terdiri data,left,right}
        return Node(input:data)
procedure insert(root: address,input:data)
{memasukkan elemen ke node}
  if root == null then:
        return NodeBaru()
  else:
        if data < root.data then:
                root.left = insert(data,root.left)
        else:
                root.right = insert(data,root.right)
        return root
```

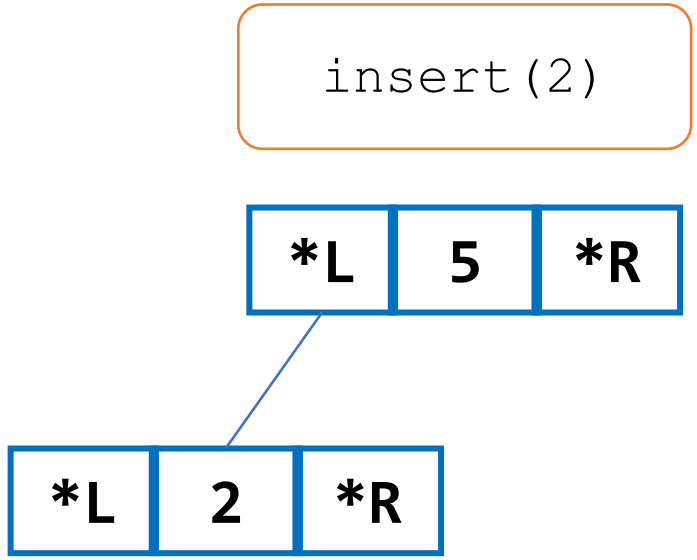
insert(5)

*L	5	*R

Kondisi awal:
root masih **null**, maka
5 menjadi node pertama
Dari binary tree

Node Baru Binary Tree

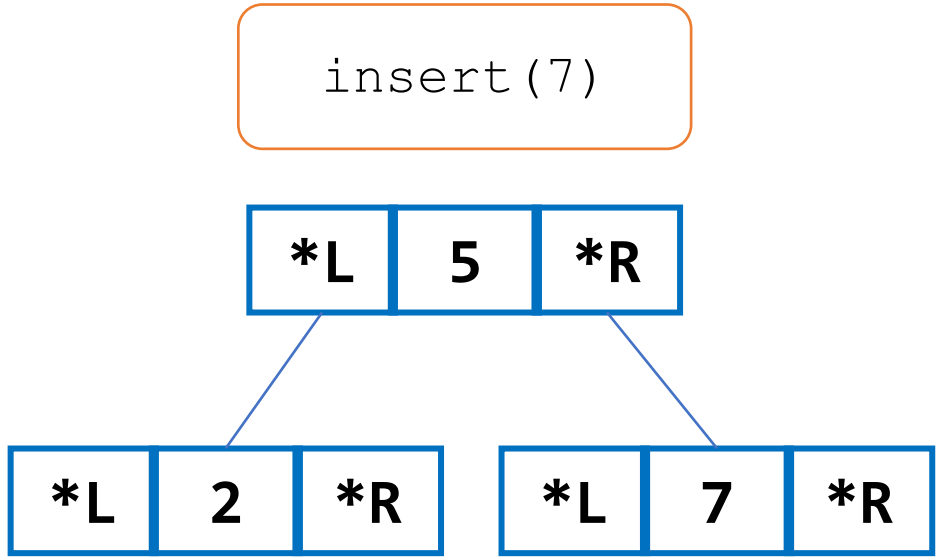
```
{definisi binary tree}
{buat node baru}
type BinaryTree:
procedure NodeBaru(input: data)
{buat node baru terdiri data,left,right}
        return Node(input:data)
procedure insert(root: address,input:data)
{memasukkan elemen ke node}
  if root == null then:
        return NodeBaru()
  else:
        if data < root.data then:
                root.left = insert(data,root.left)
        else:
                root.right = insert(data,root.right)
        return root
```



Root tidak lagi null:
Data **lebih kecil** dari root Sebelumnya. Maka, 2 Diletakkan **node kiri**

Node Baru Binary Tree

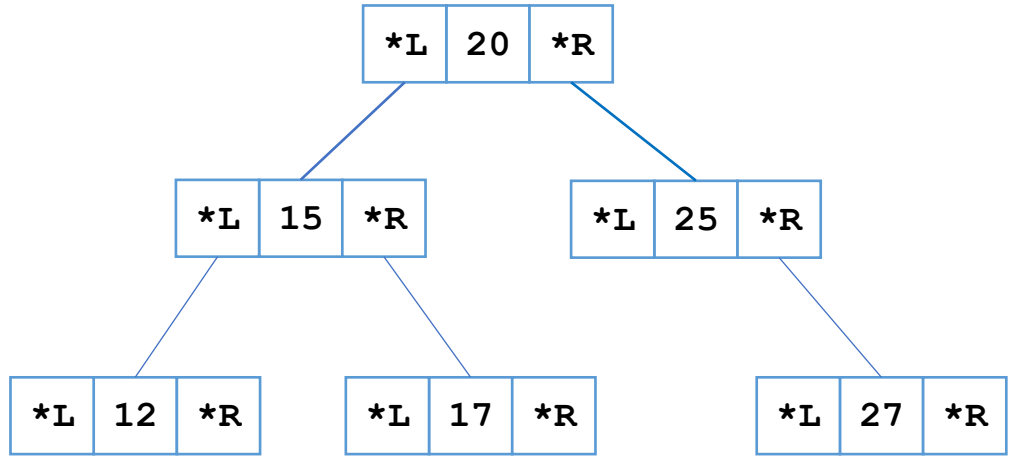
```
{definisi binary tree}
{buat node baru}
type BinaryTree:
procedure NodeBaru(input: data)
{buat node baru terdiri data,left,right}
        return Node(input:data)
procedure insert(root: address,input:data)
{memasukkan elemen ke node}
  if root == null then:
        return NodeBaru()
  else:
        if data < root.data then:
                root.left = insert(data,root.left)
        else:
                root.right = insert(data,root.right)
        return root
```



Root tidak lagi null:
Data **lebih besar** dari root Sebelumnya. Maka, 7 Diletakkan **node kanan**

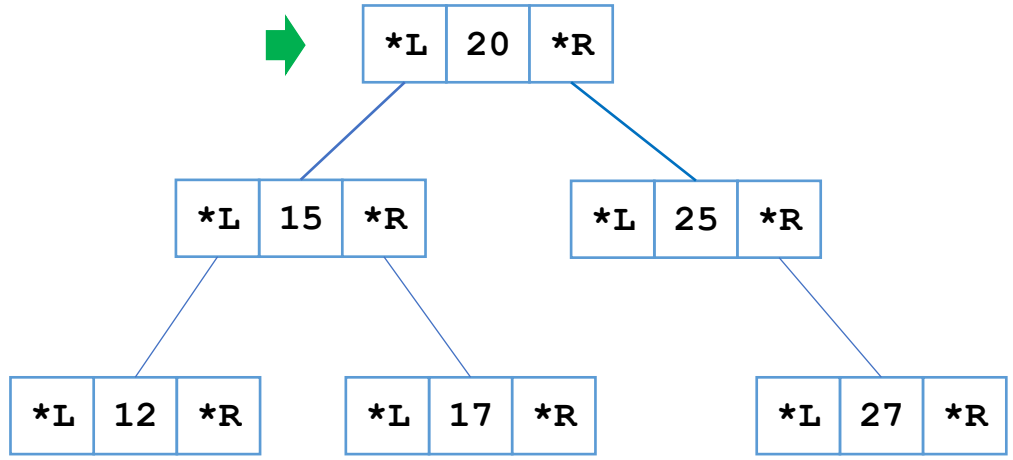
Print PreOrder

- PreOrder mencetak data dengan urutan:
- Data/Parent → Left → Right:
- Tahap:
  - Jika root tidak null:
    1. Visit root → cetak data
    2. Visit root kiri
    3. Visit root kanan



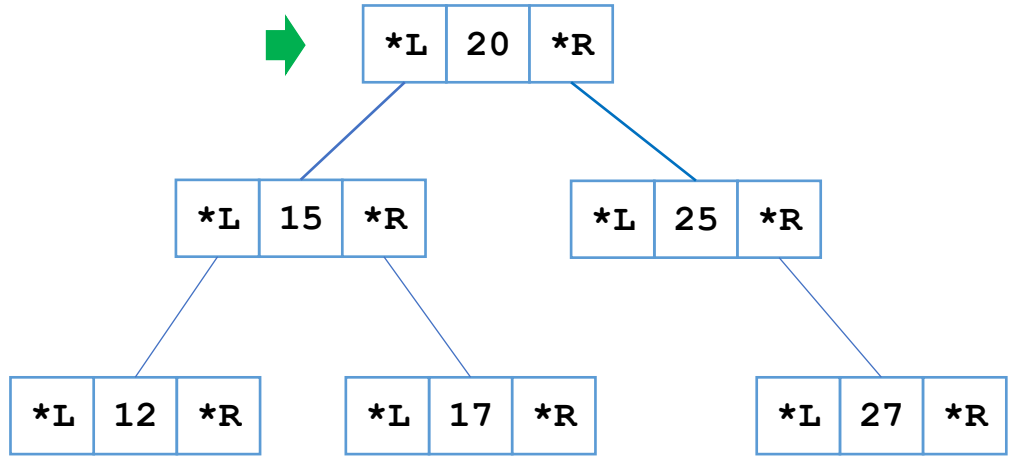
Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
      output (root.data)
      PreOrder(root.left)
      PreOrder(root.right)
```



Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



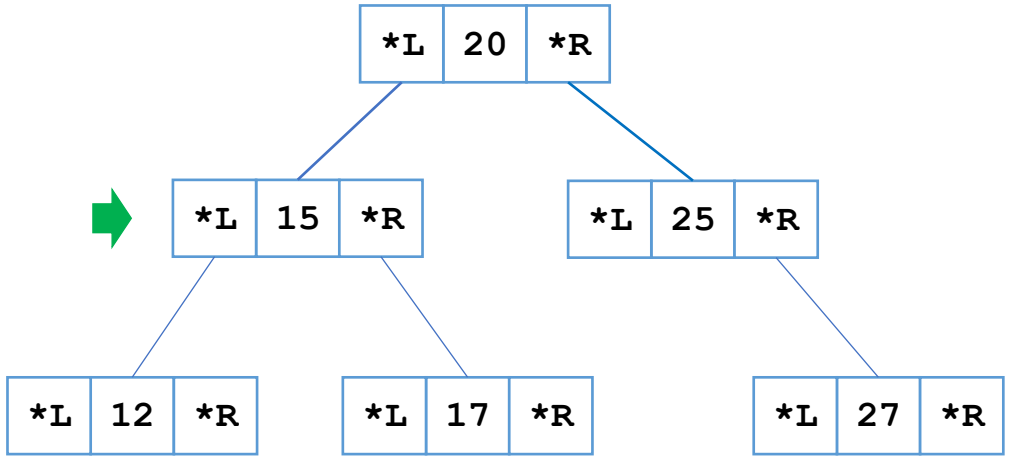
Hasil:

20

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



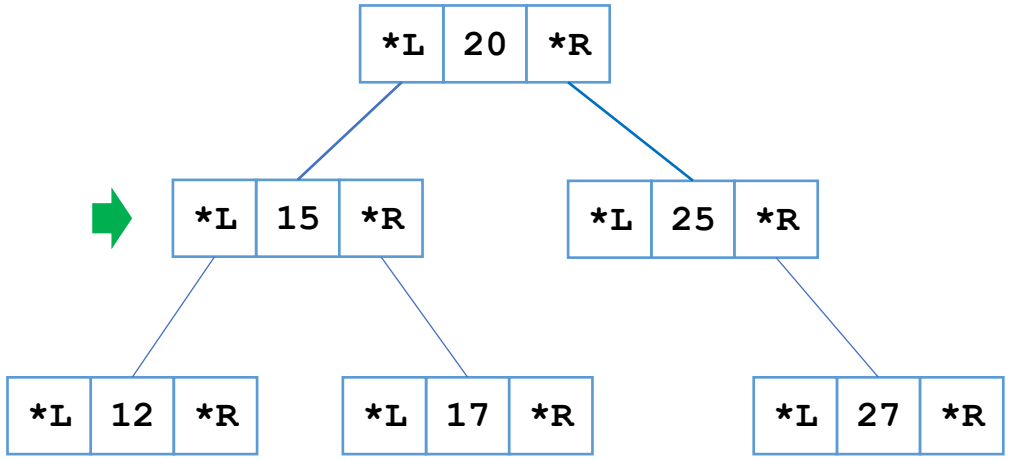
Hasil:

20

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



Hasil:

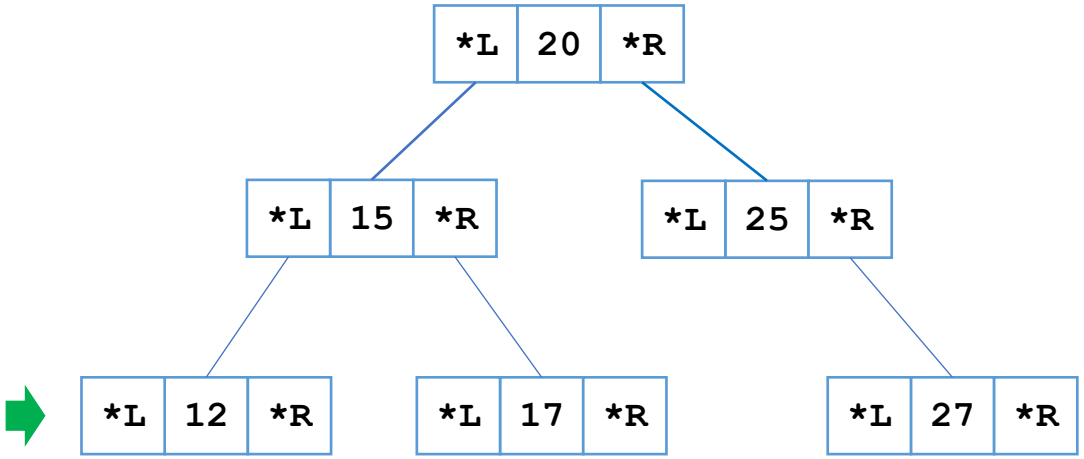
20 15

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



Hasil:

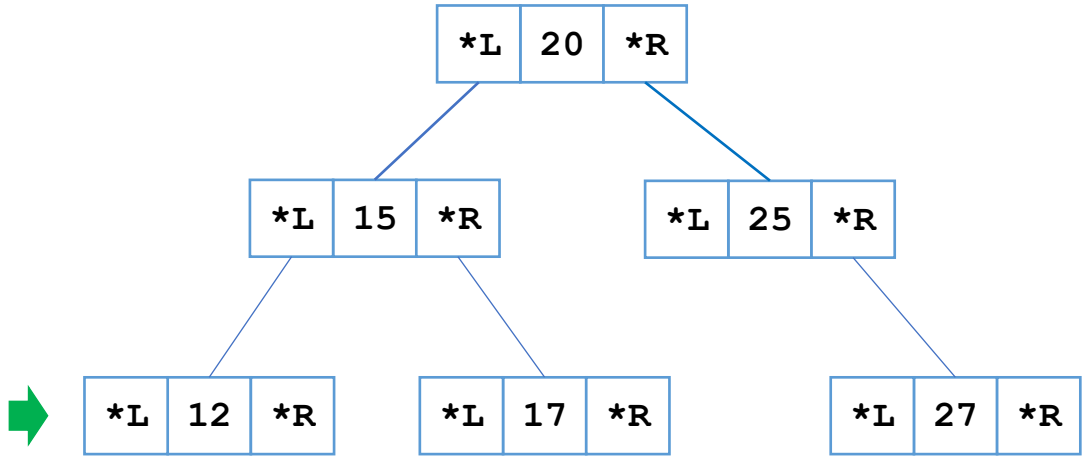
20 15

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



Hasil:

20 15 12

Ilustrasi - Notasi Algoritmik - PreOrder

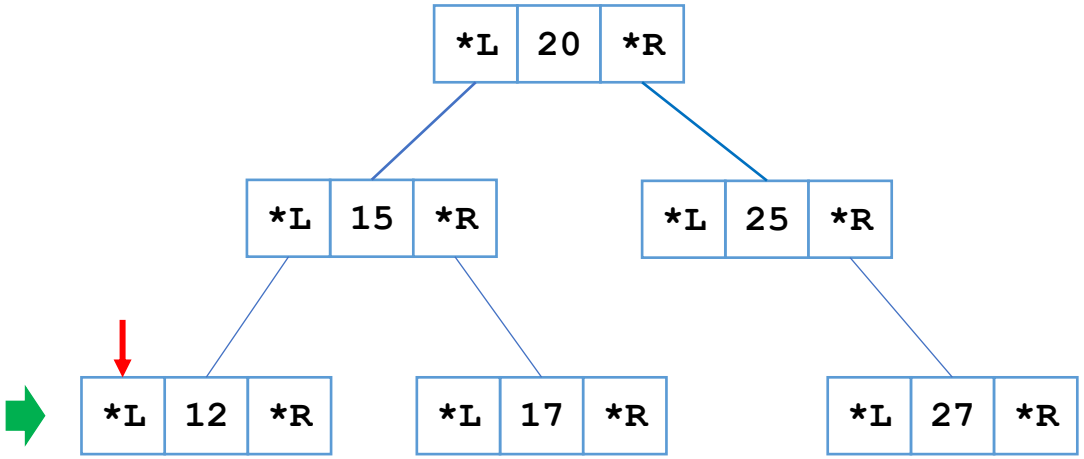
```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     ...
```

Root == null
Eksekusi Code selanjutnya



Hasil:

20 15 12

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

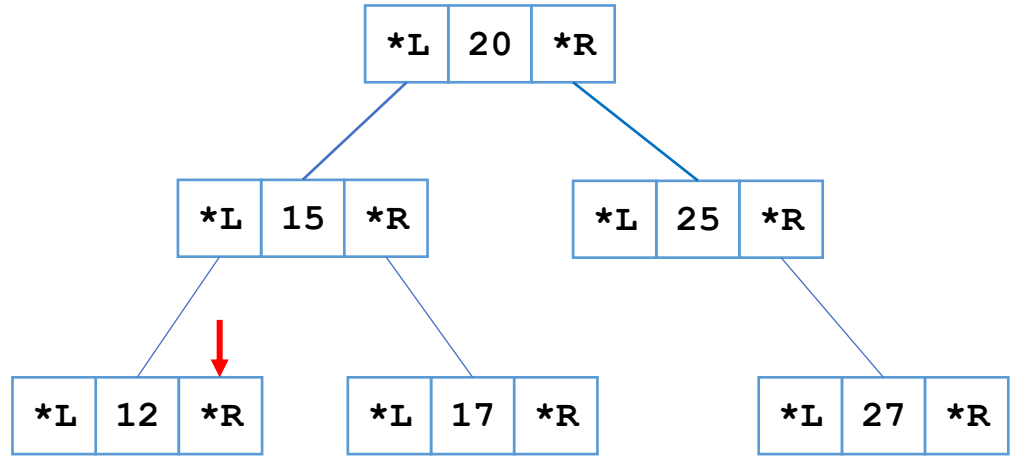
```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     ...
```

Fungsi Selesai dijalankan
Kembali ke fungsi sebelumnya
Eksekusi code selanjutnya

Root == null
Eksekusi selesai



Hasil:

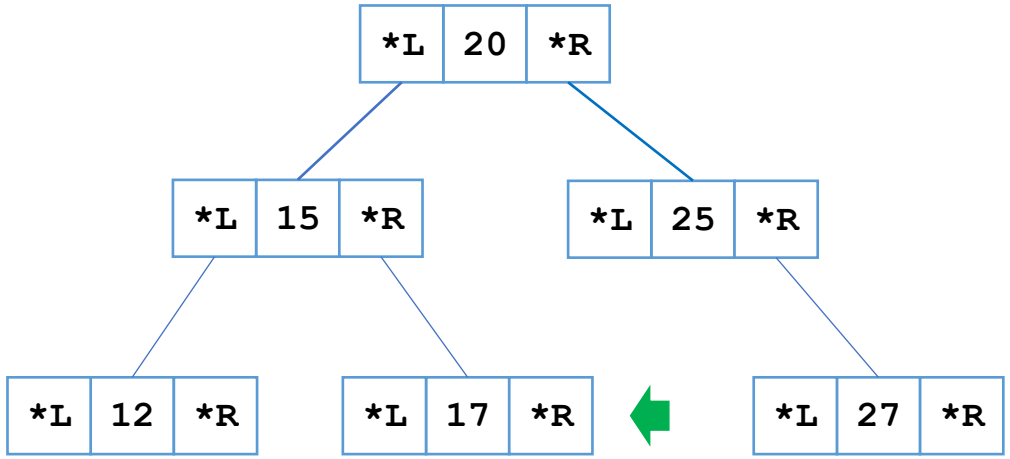
20 15 12

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



Hasil:

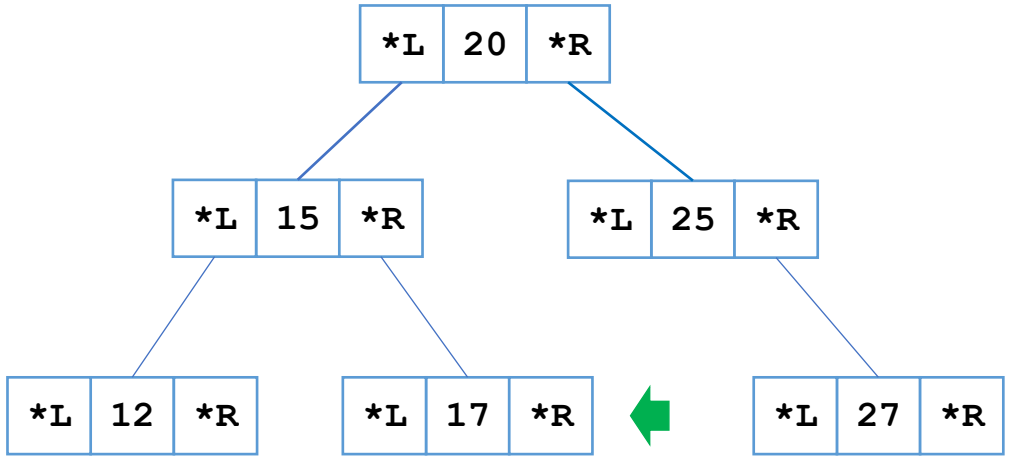
20 15 12

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



Hasil:

20 15 12 17

Ilustrasi - Notasi Algoritmik - PreOrder

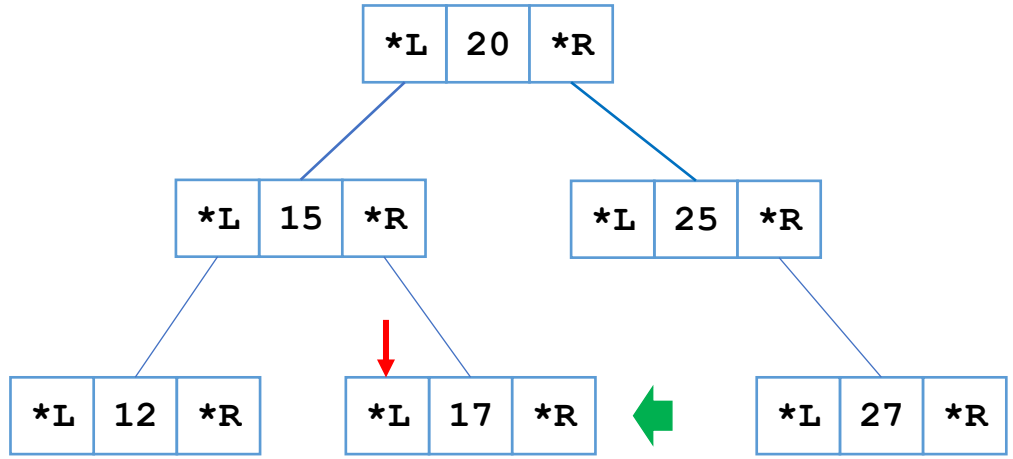
```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     . . .
```

Root == null
Eksekusi Code selanjutnya



Hasil:

20 15 12 17

Ilustrasi - Notasi Algoritmik - PreOrder

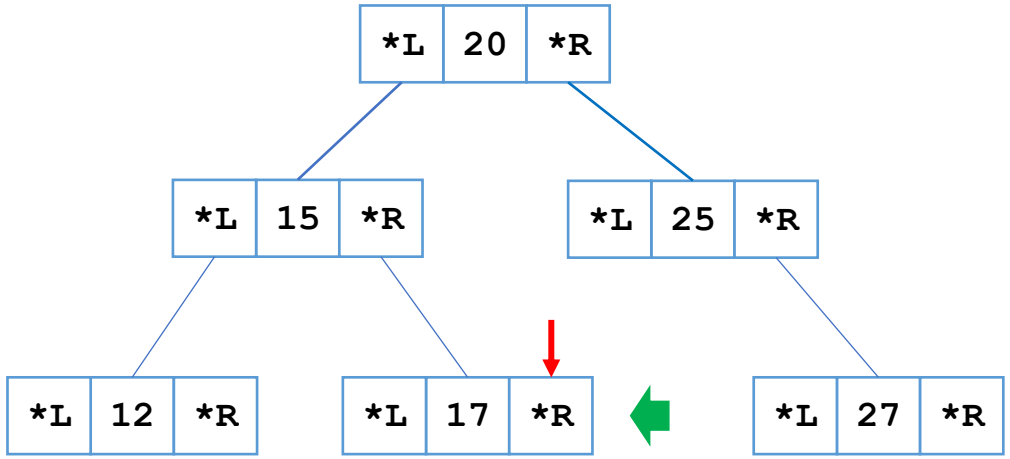
```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     . . .
```

Root == null
Eksekusi Code selesai



Hasil:

20 15 12 17

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

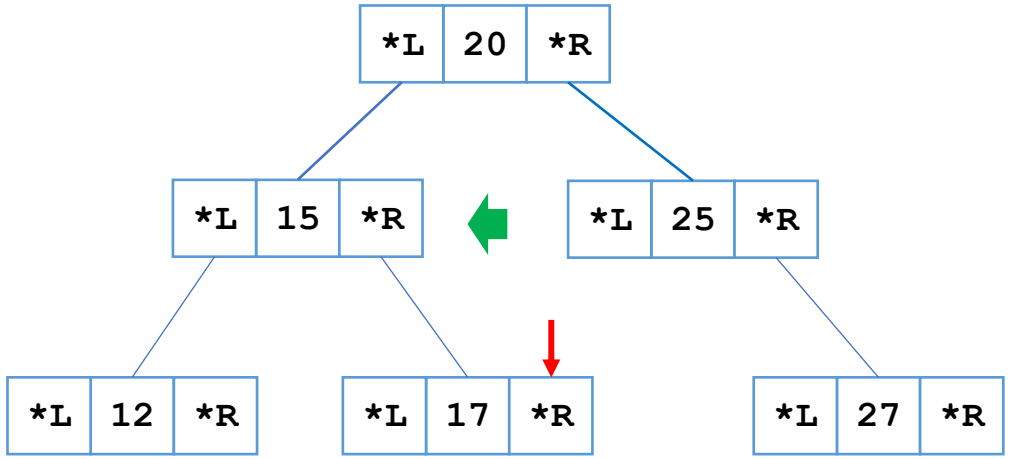
Fungsi Selesai dijalankan
Kembali ke fungsi sebelumnya
Eksekusi code selanjutnya

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

✓

```
procedure PreOrder(root:address)
  if root != null then
     . . .
```

Root == null
Eksekusi Code selesai



Hasil:

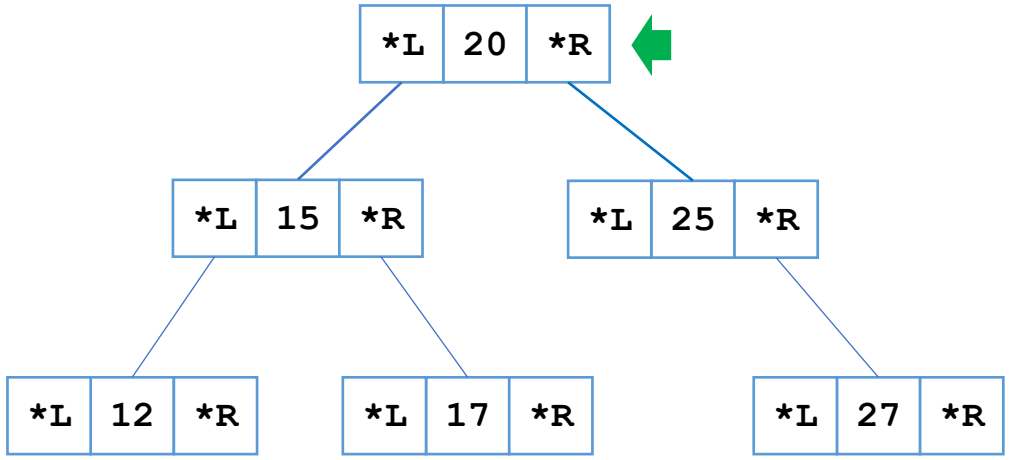
20 15 12 17

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

Fungsi Selesai dijalankan
Kembali ke fungsi sebelumnya
Eksekusi code selanjutnya



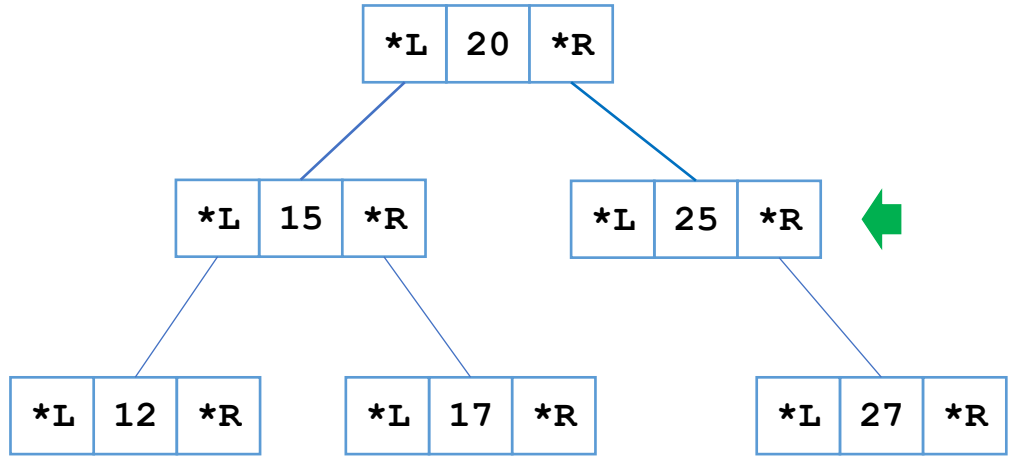
Hasil:

20 15 12 17

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



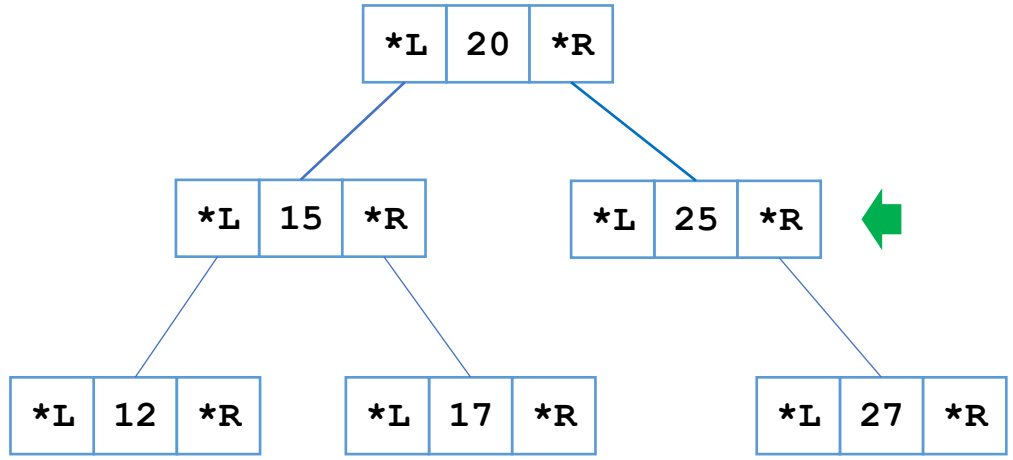
Hasil:

20 15 12 17

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



Hasil:

20 15 12 17 25

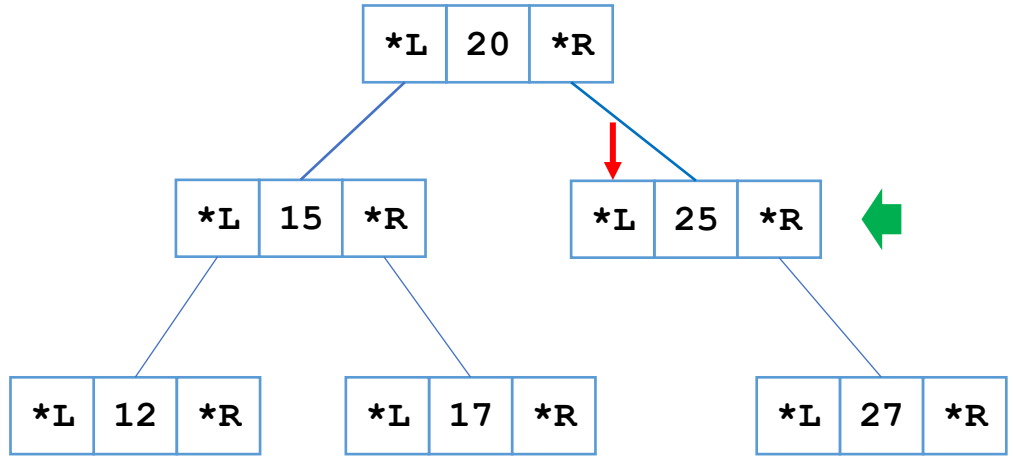
Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

Root == null
Eksekusi Code selanjutnya



Hasil:

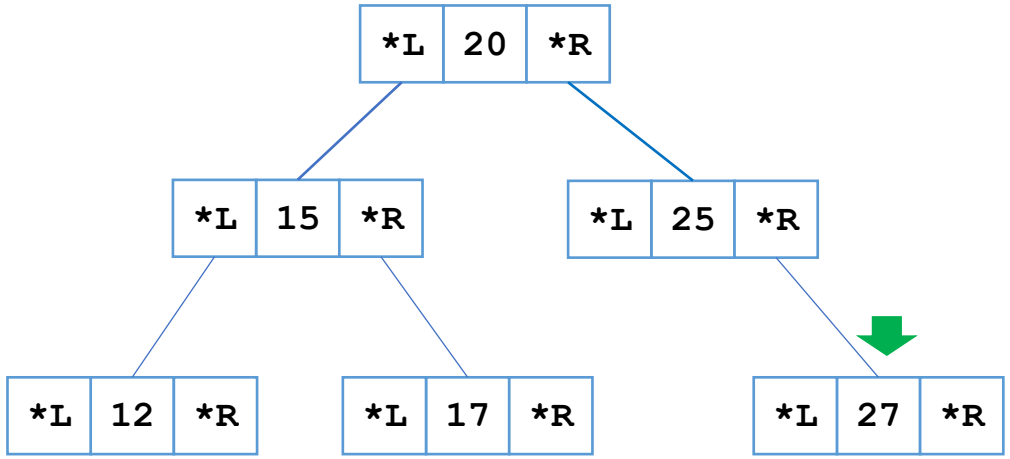
20 15 12 17 25

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



Hasil:

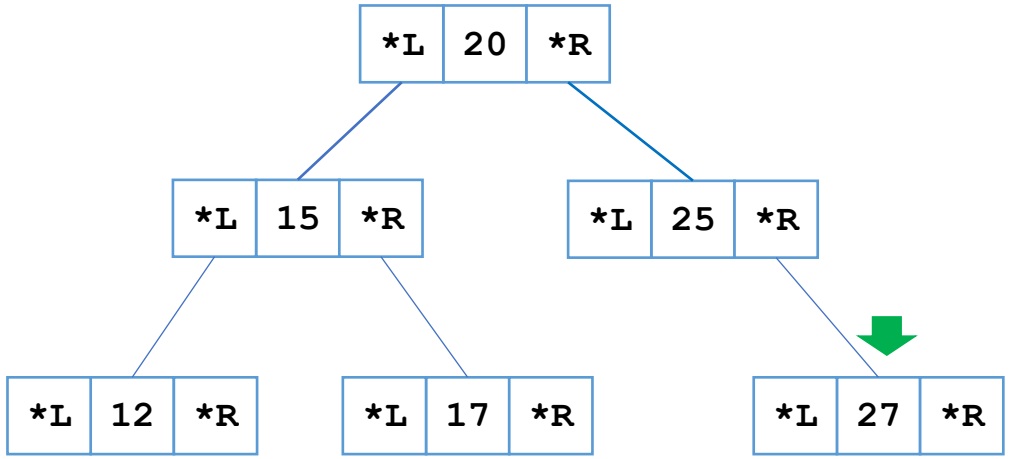
20 15 12 17 25

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```



Hasil:

20 15 12 17 25 27

Ilustrasi - Notasi Algoritmik - PreOrder

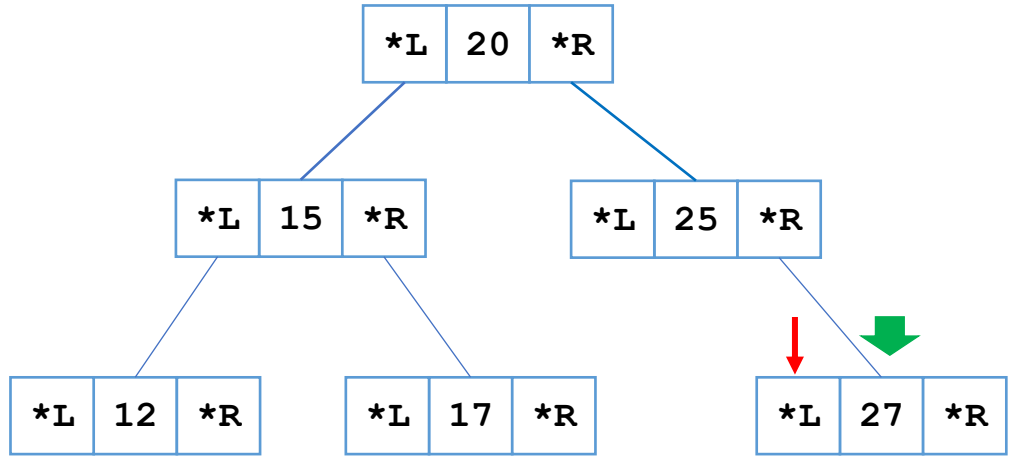
```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     . . .
```

Root == null
Eksekusi Code selanjutnya



Hasil:

20 15 12 17 25 27

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

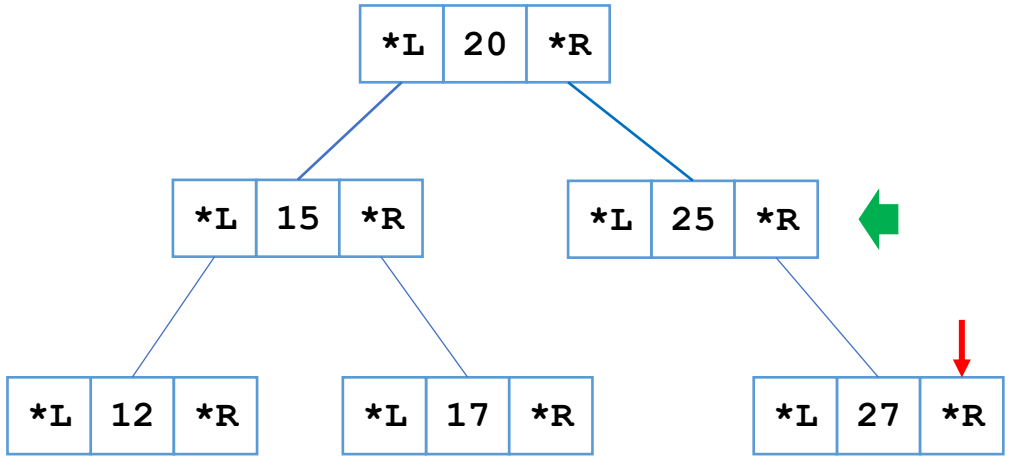
Fungsi Selesai dijalankan
Kembali ke fungsi sebelumnya
Eksekusi code selanjutnya

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

✓

```
procedure PreOrder(root:address)
  if root != null then
     . . .
```

Root == null
Eksekusi Code
selesai



Hasil:

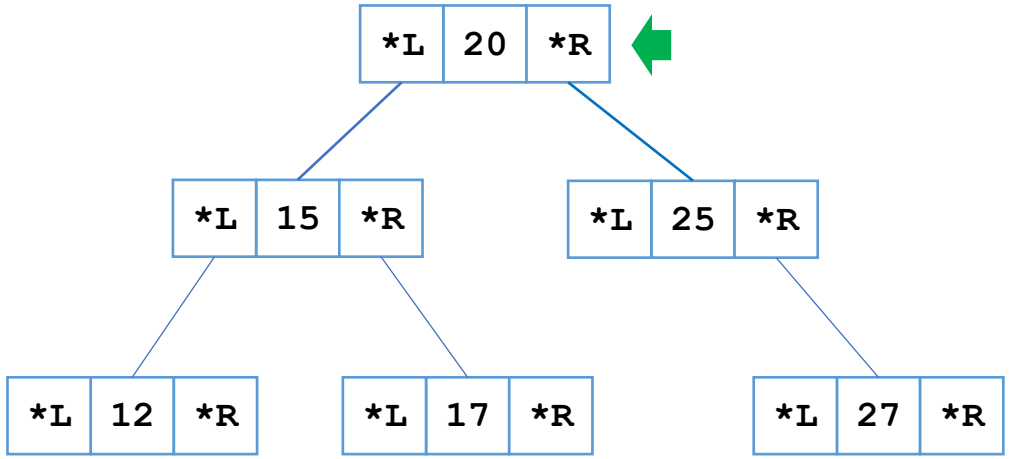
20 15 12 17 25 27

Ilustrasi - Notasi Algoritmik - PreOrder

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

Fungsi Selesai dijalankan
Kembali ke fungsi sebelumnya
Eksekusi code selanjutnya

```
procedure PreOrder(root:address)
  if root != null then
     output (root.data)
     PreOrder(root.left)
     PreOrder(root.right)
```

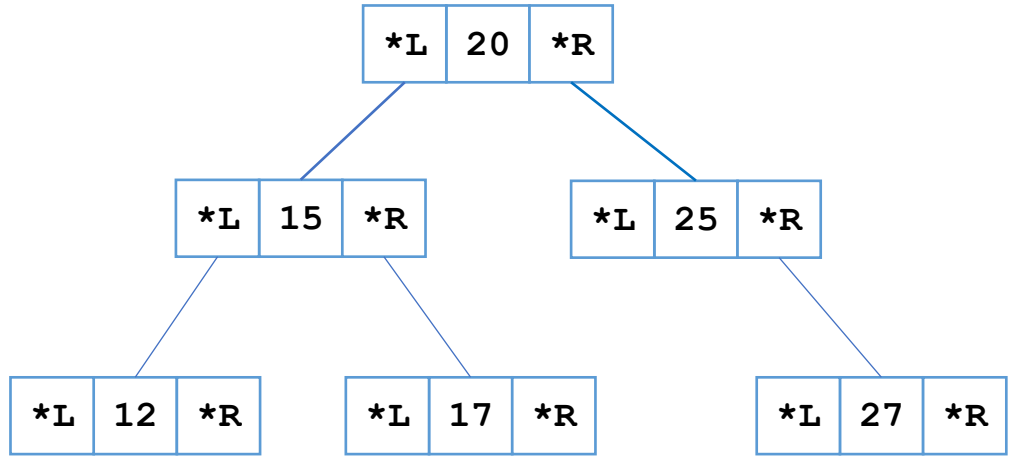


Hasil:

20 15 12 17 25 27

Ilustrasi - Notasi Algoritmik - InOrder

```
procedure InOrder(root:address)
  if root != null then
     InOrder(root.left)
     output (root.data)
     InOrder(root.right)
```

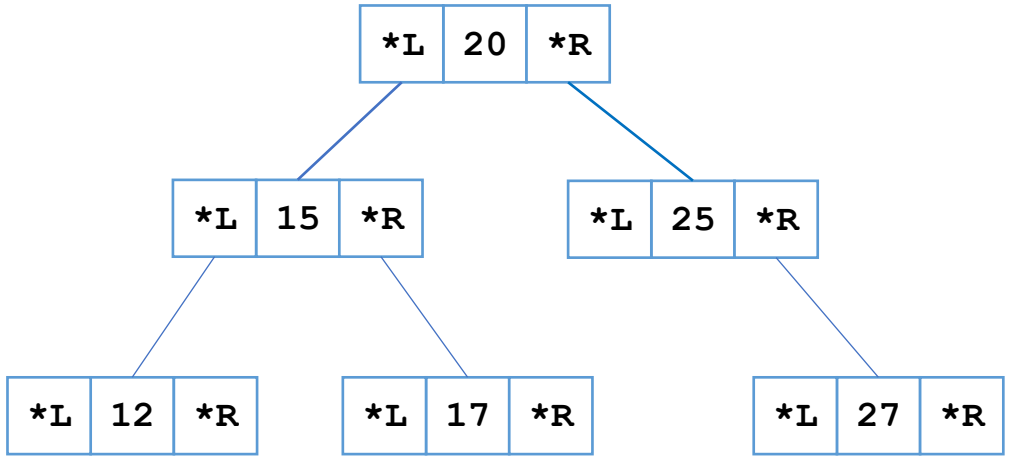


Hasil:

12 15 17 20 25 27

Ilustrasi - Notasi Algoritmik - PostOrder

```
procedure PostOrder(root:address)
  if root != null then
     PostOrder(root.left)
     PostOrder(root.right)
     output (root.data)
```



Hasil:

12 17 15 27 25 20

TERIMA KASIH

ANY QUESTIONS?