



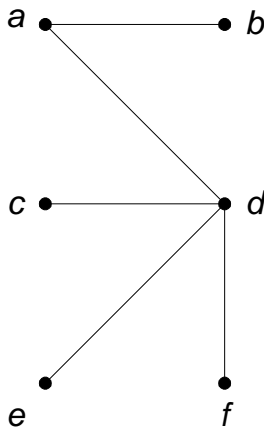
# POHON

Deasy Sandhya Elya Ikawati, S. Si, M. Si

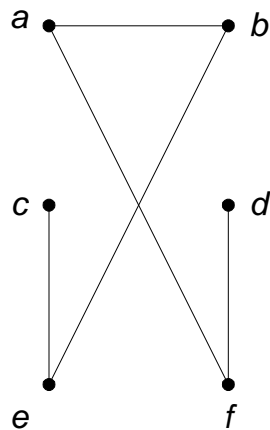
Matematika Informatika  
Politeknik Negeri Malang  
2020

# DEFINISI

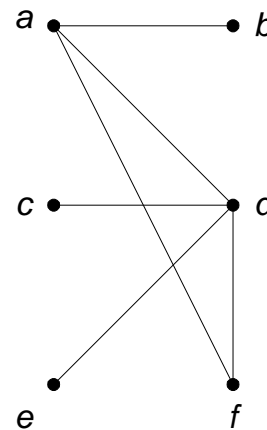
- **Pohon** adalah graf tak-berarah terhubung yang tidak mengandung sirkuit



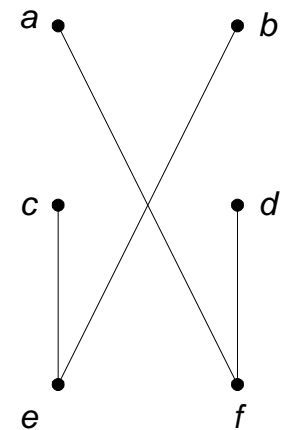
pohon



pohon



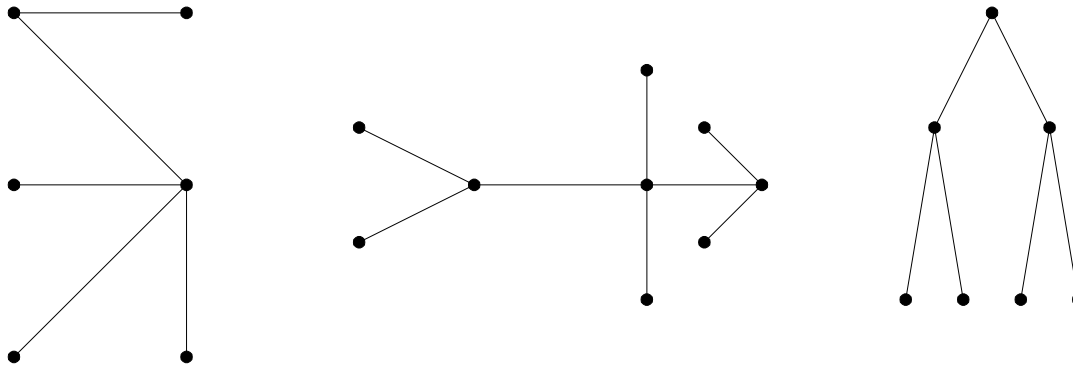
bukan pohon



bukan pohon

**Hutan** (*forest*) adalah

- kumpulan pohon yang saling lepas, atau
- graf tidak terhubung yang tidak mengandung sirkuit. Setiap komponen di dalam graf terhubung tersebut adalah pohon.



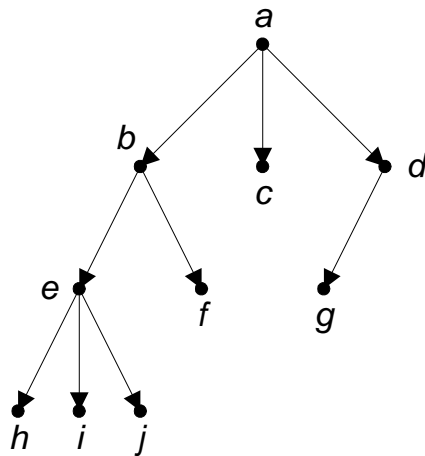
Hutan yang terdiri dari tiga buah pohon

# SIFAT-SIFAT (PROPERTI) POHON

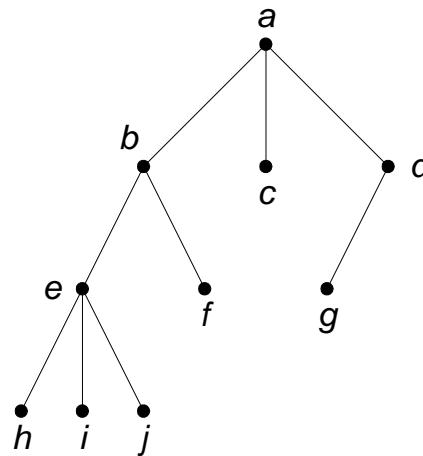
- **Teorema.** Misalkan  $G = (V, E)$  adalah graf tak-berarah sederhana dan jumlah simpulnya  $n$ . Maka, semua pernyataan di bawah ini adalah ekuivalen:
  1.  $G$  adalah pohon.
  2. Setiap pasang simpul di dalam  $G$  terhubung dengan lintasan tunggal.
  3.  $G$  terhubung dan memiliki  $m = n - 1$  buah sisi.
  4.  $G$  tidak mengandung sirkuit dan memiliki  $m = n - 1$  buah sisi.
  5.  $G$  tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat hanya satu sirkuit.
  6.  $G$  terhubung dan semua sisinya adalah jembatan.
- Teorema di atas dapat dikatakan sebagai definisi lain dari pohon.

# POHON BERAKAR (*ROOTED TREE*)

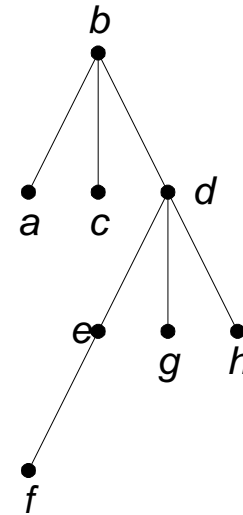
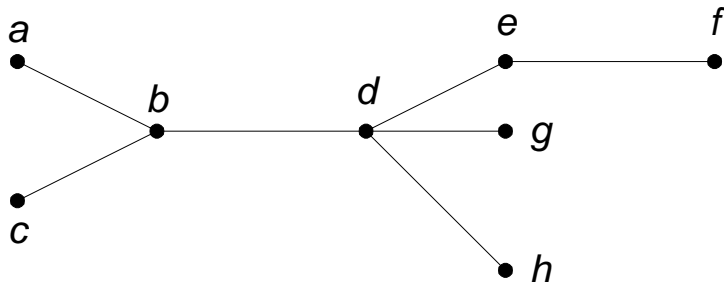
- Pohon yang satu buah simpulnya diperlakukan sebagai akar dan sisi-sisinya diberi arah sehingga menjadi graf berarah dinamakan **pohon berakar** (*rooted tree*).



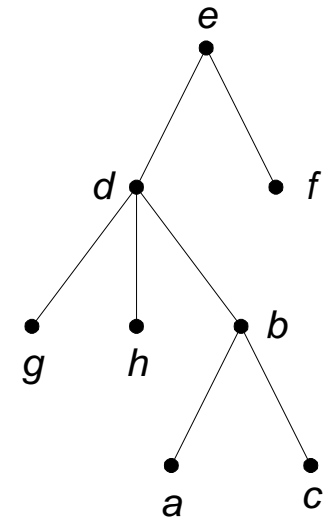
(a) Pohon berakar



(b) sebagai perjanjian, tanda panah pada sisi dapat dibuang



*b* sebagai akar



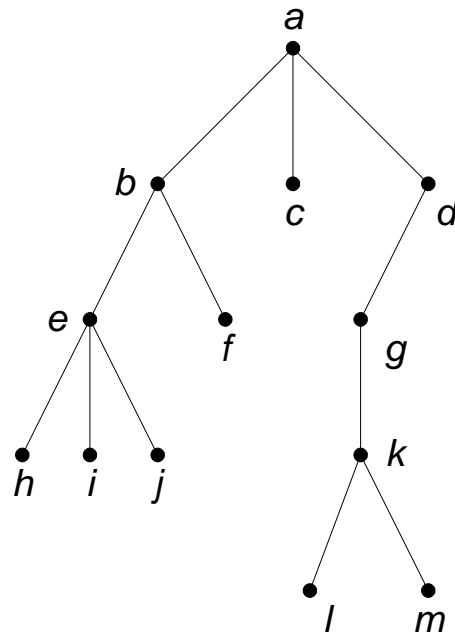
*e* sebagai akar

Pohon dan dua buah pohon berakar yang dihasilkan dari pemilihan dua simpul berbeda sebagai akar

# TERMINOLOGI PADA POHON BERAKAR

## Anak (*child* atau *children*) dan Orangtua (*parent*)

$b$ ,  $c$ , dan  $d$  adalah anak-anak simpul  $a$ ,  
 $a$  adalah orangtua dari anak-anak itu



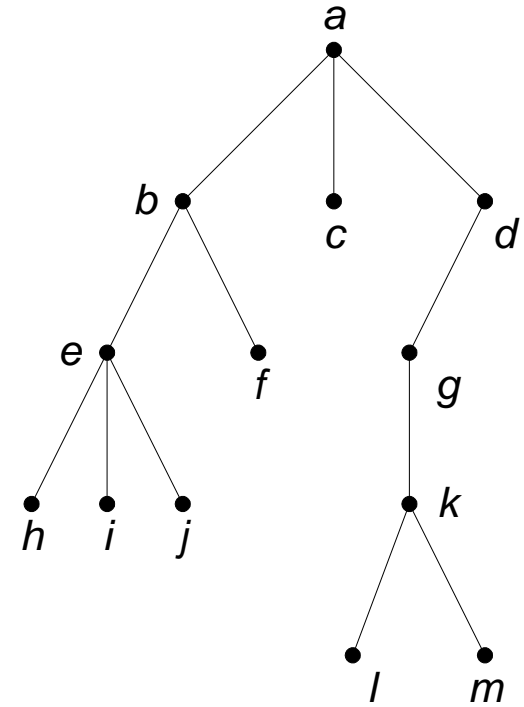
## 2. Lintasan (*path*)

Lintasan dari  $a$  ke  $j$  adalah  $a, b, e, j$ .

Panjang lintasan dari  $a$  ke  $j$  adalah 3.

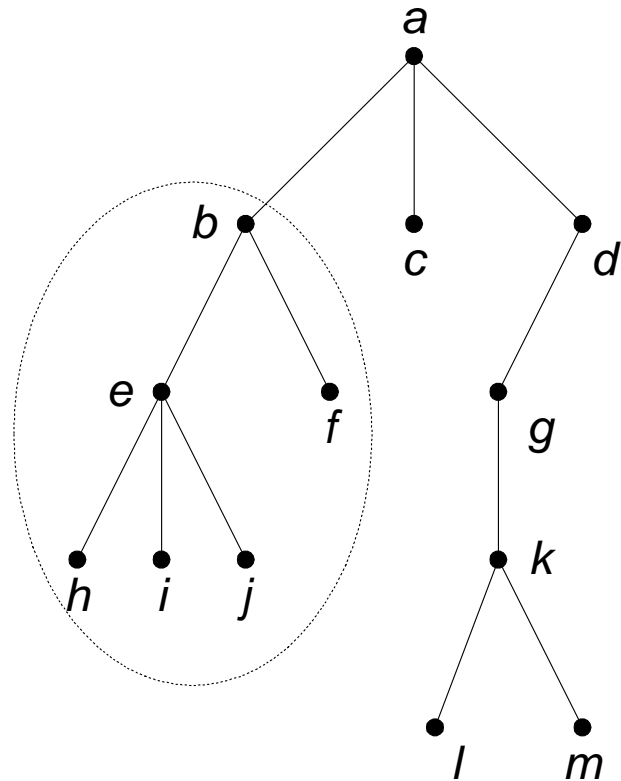
## 3. Saudara kandung (*sibling*)

$f$  adalah saudara kandung  $e$ , tetapi  $g$  bukan saudara kandung  $e$ , karena orangtua mereka berbeda.





## 4. Upapohon (*subtree*)



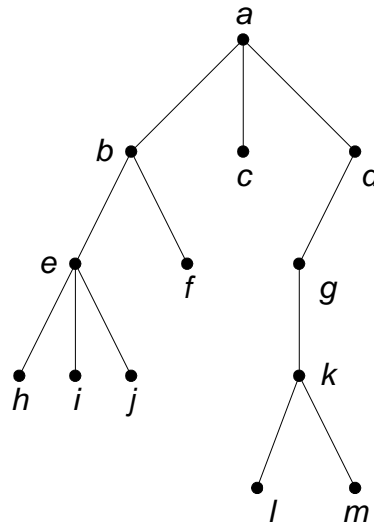
## 5. Derajat (*degree*)

**Derajat** sebuah simpul adalah jumlah upapohon (atau jumlah anak) pada simpul tersebut.

Derajat  $a$  adalah 3, derajat  $b$  adalah 2,  
Derajat  $d$  adalah satu dan derajat  $c$  adalah 0.

Jadi, derajat yang dimaksudkan di sini adalah derajat-keluar.

Derajat maksimum dari semua simpul merupakan derajat pohon itu sendiri. Pohon di atas berderajat 3

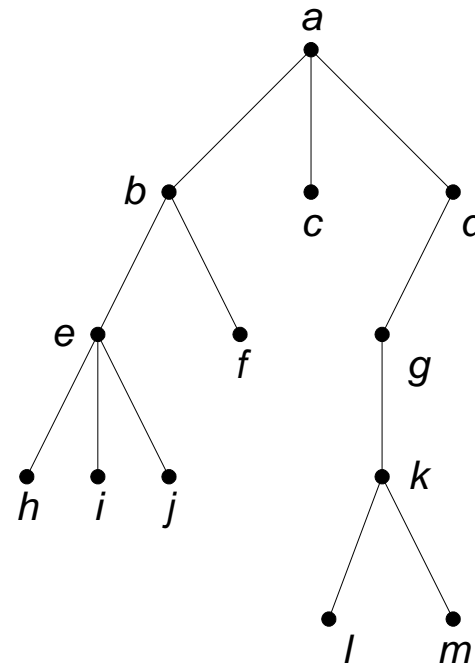


## 6. Daun (*leaf*)

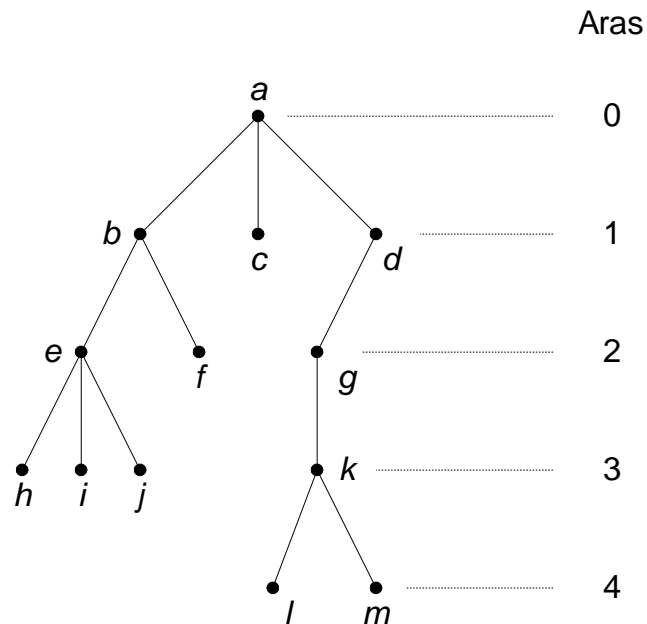
Simpul yang berderajat nol (atau tidak mempunyai anak) disebut **daun**. Simpul  $h, i, j, f, c, l$ , dan  $m$  adalah daun.

## 7. Simpul Dalam (*internal nodes*)

Simpul yang mempunyai anak disebut **simpul dalam**. Simpul  $b, d, e, g$ , dan  $k$  adalah simpul dalam.



## 8. Aras (*level*) atau Tingkat

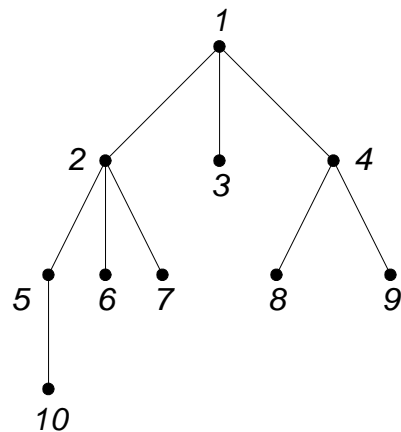


## 9. Tinggi (*height*) atau Kedalaman (*depth*)

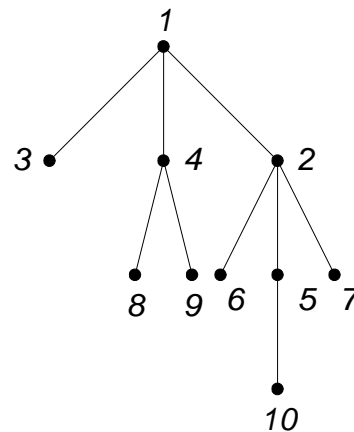
Aras maksimum dari suatu pohon disebut **tinggi** atau **kedalaman** pohon tersebut. Pohon di atas mempunyai tinggi 4.

# POHON TERURUT (*ORDERED TREE*)

Pohon berakar yang urutan anak-anaknya penting disebut **pohon terurut** (*ordered tree*).



(a)

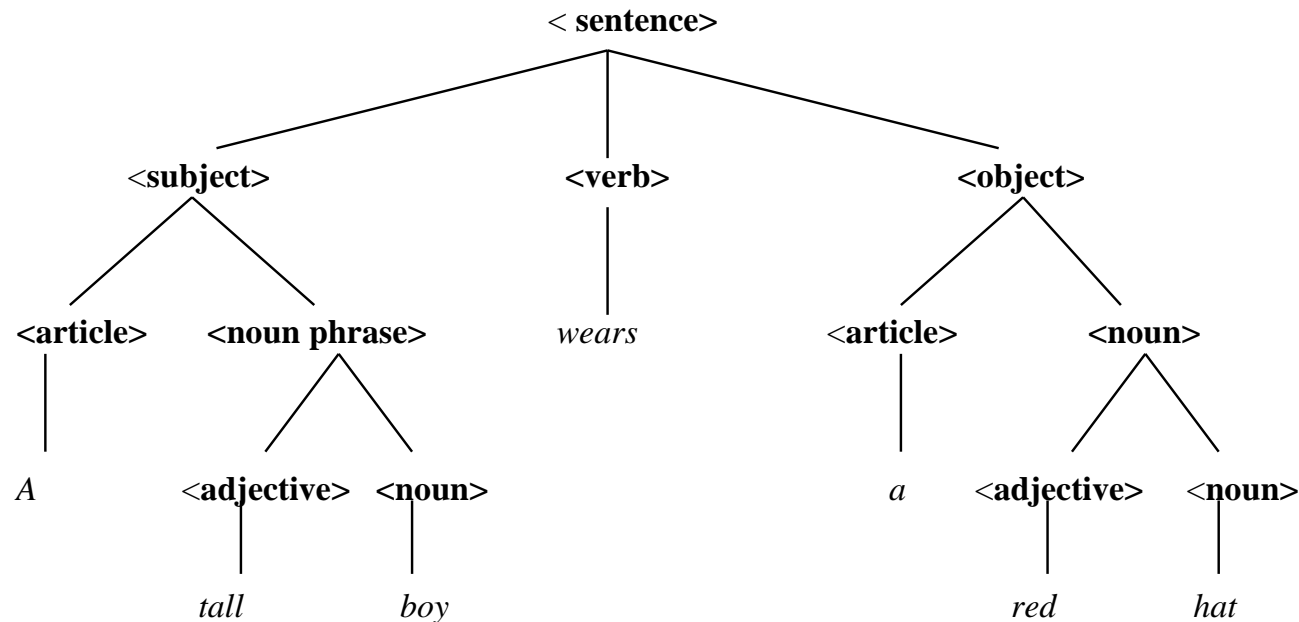


(b)

(a) dan (b) adalah dua pohon terurut yang berbeda

# POHON *N-ARY*

- Pohon berakar yang setiap simpul cabangnya mempunyai paling banyak  $n$  buah anak disebut **pohon  $n$ -ary**.

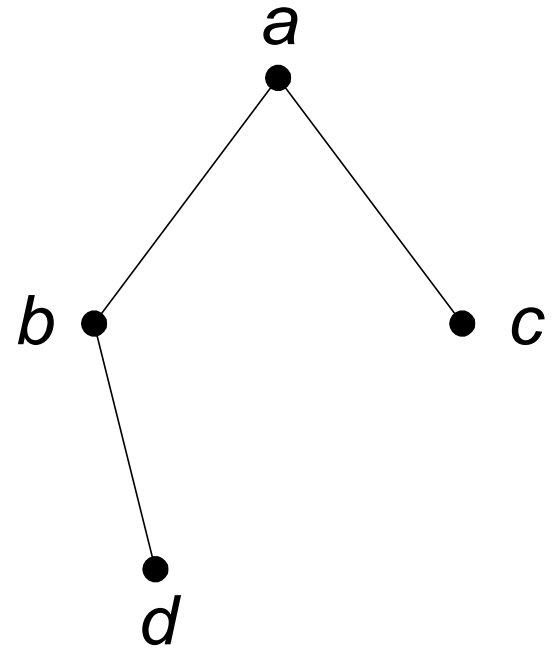
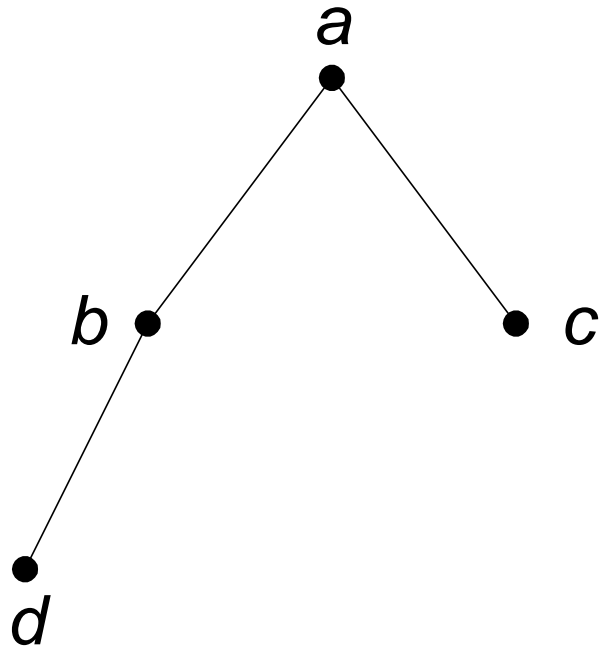


**Gambar** Pohon parsing dari kalimat *A tall boy wears a red hat*

- Pohon  $n$ -ary dikatakan **teratur** atau **penuh** (*full*) jika setiap simpul cabangnya mempunyai tepat  $n$  anak.

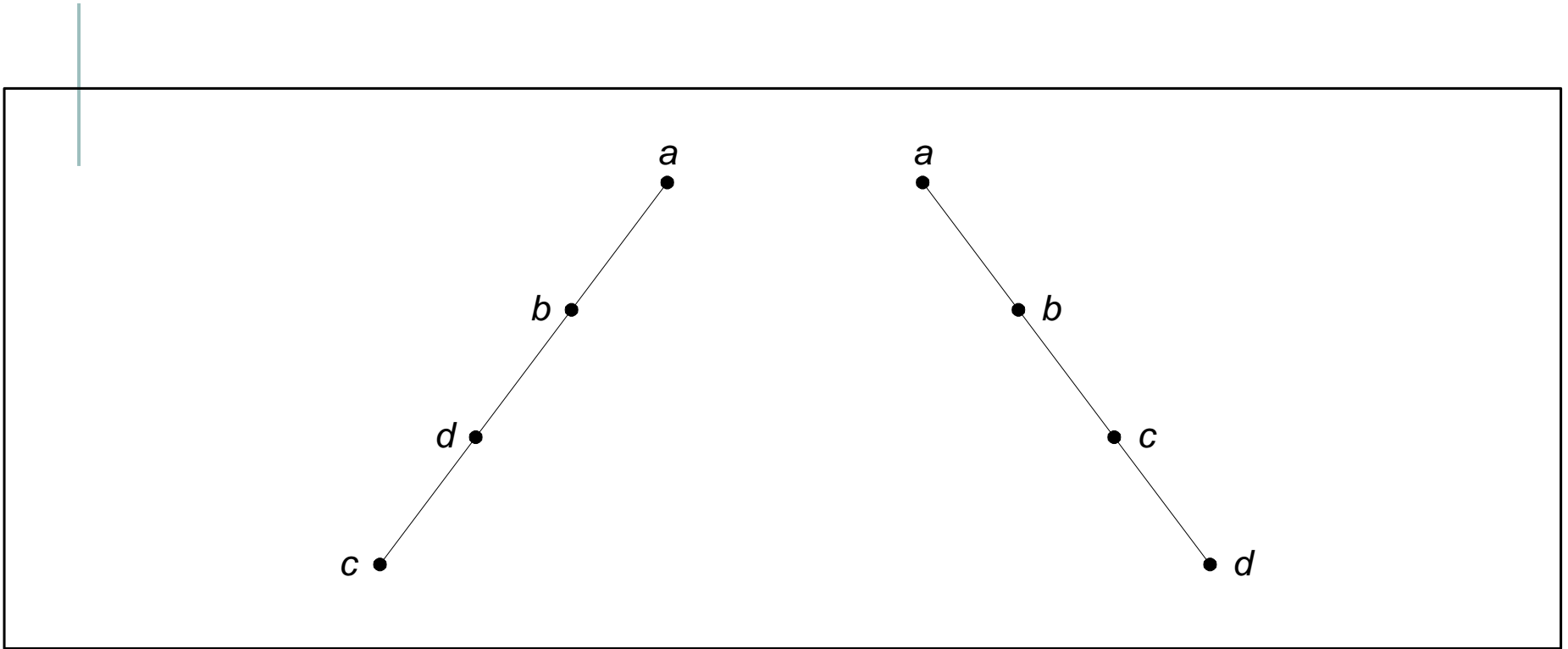
# POHON BINER (*BINARY TREE*)

- ❑ Adalah pohon  $n$ -ary dengan  $n = 2$ .
- ❑ Pohon yang paling penting karena banyak aplikasinya.
- ❑ Setiap simpul di dalam pohon biner mempunyai paling banyak 2 buah anak.
- ❑ Dibedakan antara anak kiri (*left child*) dan anak kanan (*right child*)
- ❑ Karena ada perbedaan urutan anak, maka pohon biner adalah pohon terurut.

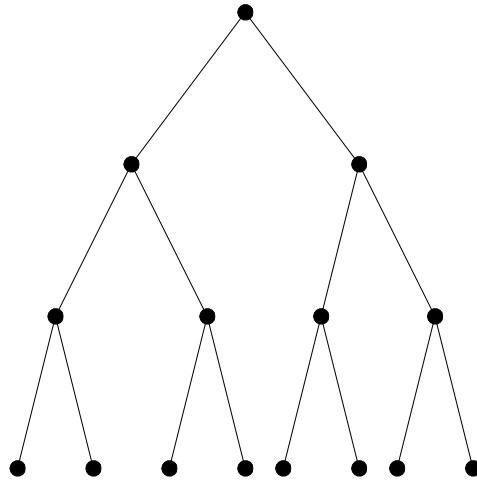


**Gambar** Dua buah pohon biner yang berbeda





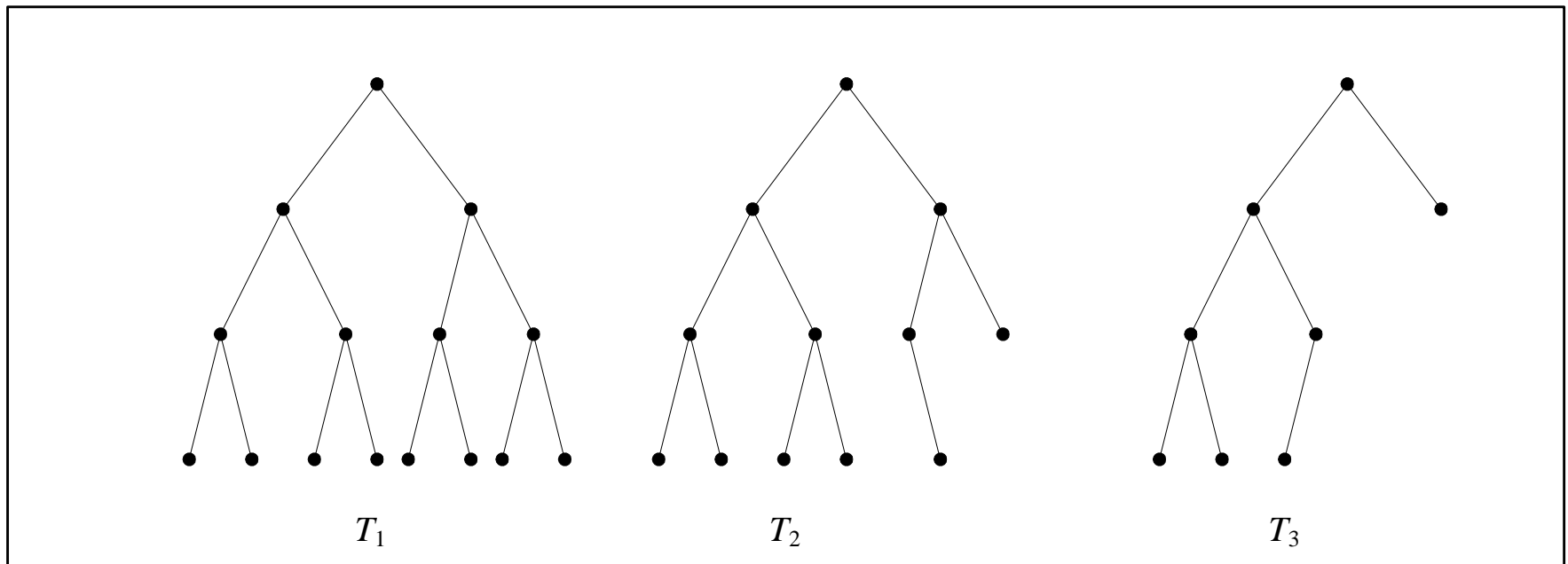
**Gambar** (a) Pohon condong-kiri, dan (b) pohon condong kanan



**Gambar** Pohon biner penuh

# Pohon Biner Seimbang

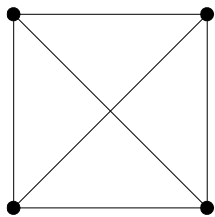
Pada beberapa aplikasi, diinginkan tinggi upapohon kiri dan tinggi upapohon kanan yang seimbang, yaitu berbeda maksimal 1.



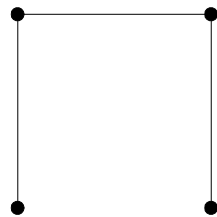
**Gambar**  $T_1$  dan  $T_2$  adalah pohon seimbang, sedangkan  $T_3$  bukan pohon seimbang.

# POHON MERENTANG (*SPANNING TREE*)

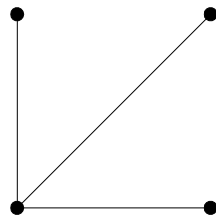
- Pohon merentang dari graf terhubung adalah upagraf merentang yang berupa pohon.
- Pohon merentang diperoleh dengan memutus sirkuit di dalam graf.



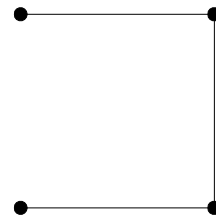
$G$



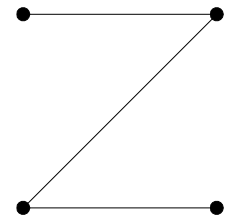
$T_1$



$T_2$



$T_3$

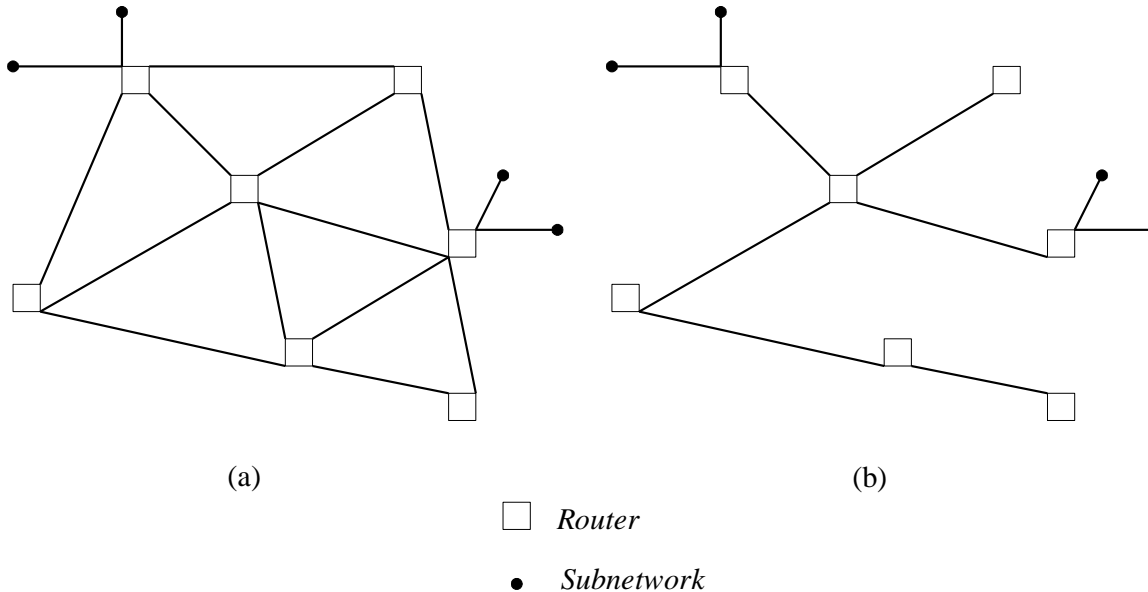


$T_4$

- Setiap graf terhubung mempunyai paling sedikit satu buah pohon merentang.
- Graf tak-terhubung dengan  $k$  komponen mempunyai  $k$  buah hutan merentang yang disebut hutan merentang (*spanning forest*).

# APLIKASI POHON MERENTANG

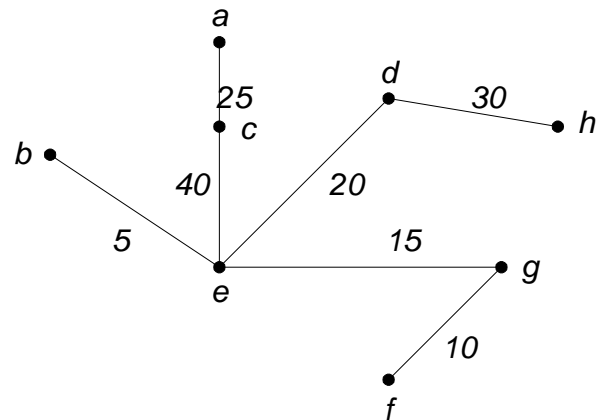
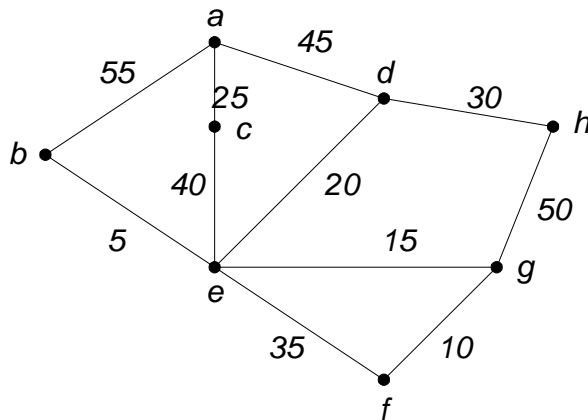
1. Jumlah ruas jalan seminimum mungkin yang menghubungkan semua kota sehingga setiap kota tetap terhubung satu sama lain.
2. Perutean (*routing*) pesan pada jaringan komputer.



(a) Jaringan komputer, (b) Pohon merentang *multicast*

# POHON MERENTANG MINIMUM

- Graf terhubung-berbobot mungkin mempunyai lebih dari 1 pohon merentang.
- Pohon merentang yang berbobot minimum –dinamakan **pohon merentang minimum** (*minimum spanning tree*).



# ALGORITMA PRIMS

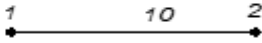
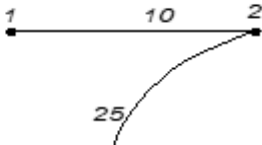
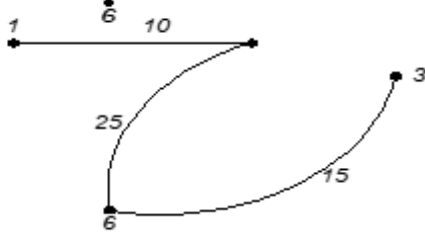
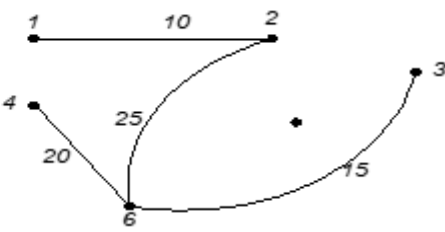
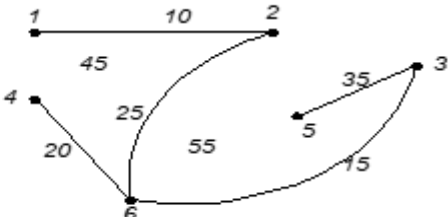
**Langkah 1 :** Ambil sisi graf  $G$  yang berbobot minimum, masukkan ke dalam  $T$ .

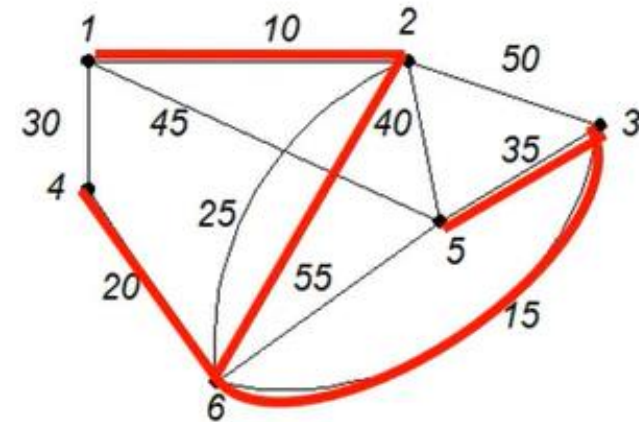
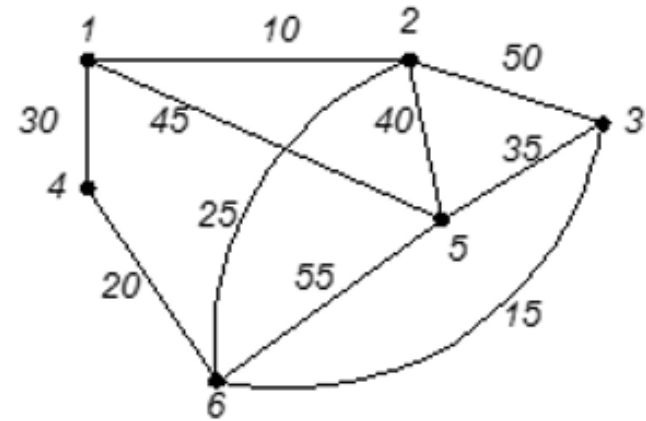
**Langkah 2 :** Pilih sisi  $(u,v)$  yang mempunyai bobot minimum dan bersisian dengan simpul  $T$ , tetapi  $(U, v)$  tidak membentuk sirkuit di  $T$ . Masukkan  $(u, v)$  ke dalam  $T$ .

**Langkah 3 :** Ulangi langkah 2 sebanyak  $n-2$  kali.



# CONTOH ALGORITMA PRIMS

Langkah	Sisi	Bobot	Pohon rentang
1	(1, 2)	10	
2	(2, 6)	25	
3	(3, 6)	15	
4	(4, 6)	20	
5	(3, 5)	35	



$$\text{Bobot} = 10 + 25 + 15 + 20 + 35 = 105$$

# ALGORITMA KRUSKAL

**Langkah 0 :** Sisi dari graf sudah diurut secara menaik berdasarkan bobotnya- dari bobot kecil ke besar;

**Langkah 1 :**  $T$  (graf) masih kosong;

**Langkah 2 :** Pilih sisi  $(u, v)$  dengan bobot minimum yang tidak membentuk sirkuit di  $T$ . Tambahkan  $(u, v)$  ke dalam  $T$ ;

**Langkah 3 :** Ulangi langkah 2 sebanyak  $n - 1$  kali.

```
procedure Kruskal(input G : graf, output T : pohon)
{ Membentuk pohon merentang minimum T dari graf terhubung -
berbobot G.
```

Masukan: graf-berbobot terhubung  $G = (V, E)$ , dengan  $|V| = n$

Keluaran: pohon rentang minimum  $T = (V, E')$

```
}
```

### **Deklarasi**

```
  i, p, q, u, v : integer
```

### **Algoritma**

```
  ( Asumsi: sisi-sisi dari graf sudah diurut menaik
    berdasarkan bobotnya - dari bobot kecil ke bobot
    besar)
```

```
  T  $\leftarrow$  {}
```

```
  while jumlah sisi T < n-1 do
```

```
    Pilih sisi (u,v) dari E yang bobotnya terkecil
```

```
    if (u,v) tidak membentuk siklus di T then
```

```
      T  $\leftarrow$  T  $\cup$  {(u,v)}
```



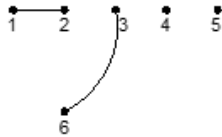
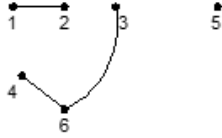
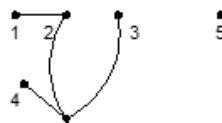
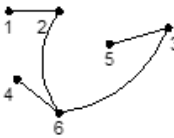
```
    endif
```

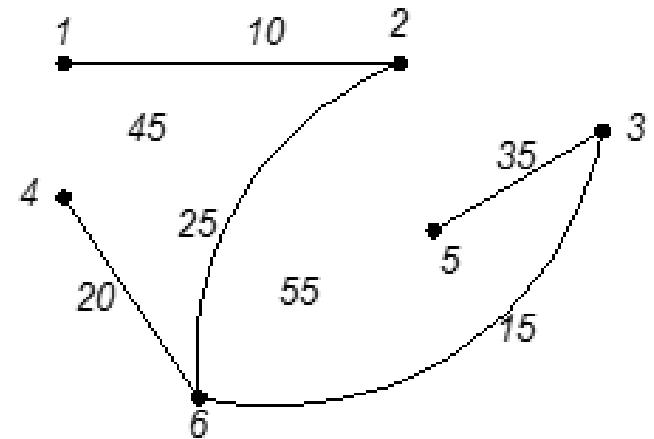
```
  endfor
```

# CONTOH ALGORITMA KRUSKAL

Sisi-sisi diurut menaik:

Sisi	(1,2)	(3,6)	(4,6)	(2,6)	(1,4)	(3,5)	(2,5)	(1,5)	(2,3)	(5,6)
Bobot	10	15	20	25	30	35	40	45	50	55

Langkah	Sisi	Bobot	Hutan merentang
0			
1	(1, 2)	10	
2	(3, 6)	15	
3	(4, 6)	20	
4	(2, 6)	25	
5	(1, 4)	30	ditolak
6	(3, 5)	35	

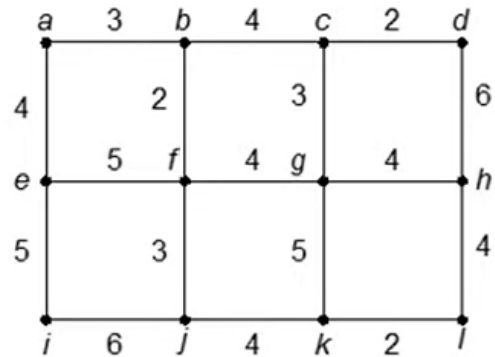


$$\text{Bobot} = 10 + 25 + 15 + 20 + 35 = 105$$

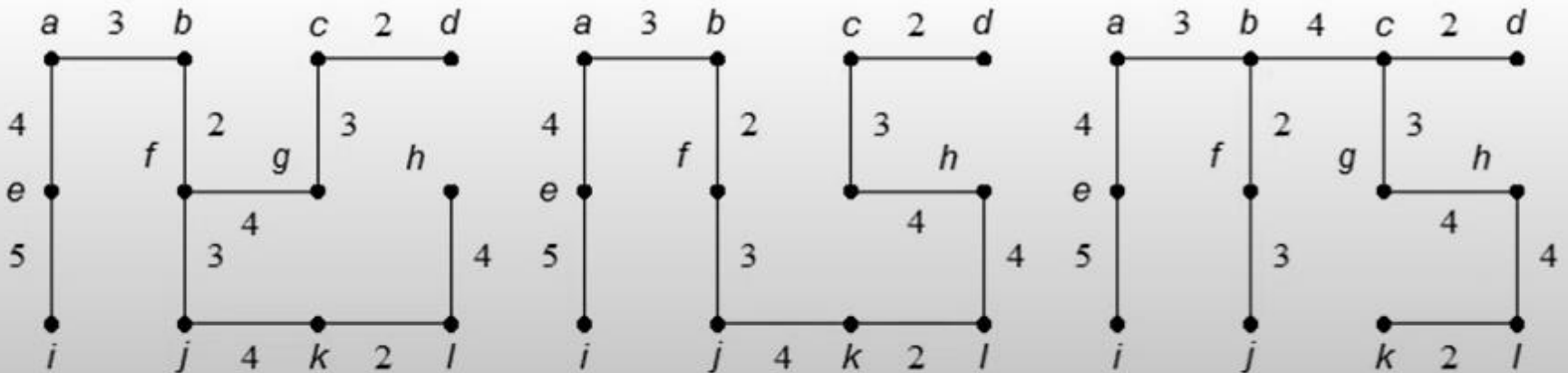
- Spanning tree yang dihasilkan tidak selalu unik, meskipun bobotnya tetap sama
- Hal ini terjadi jika ada beberapa sisi yang akan dipilih berbobot sama.

# CONTOH

Contoh :



3 buah pohon rentang minimumnya :

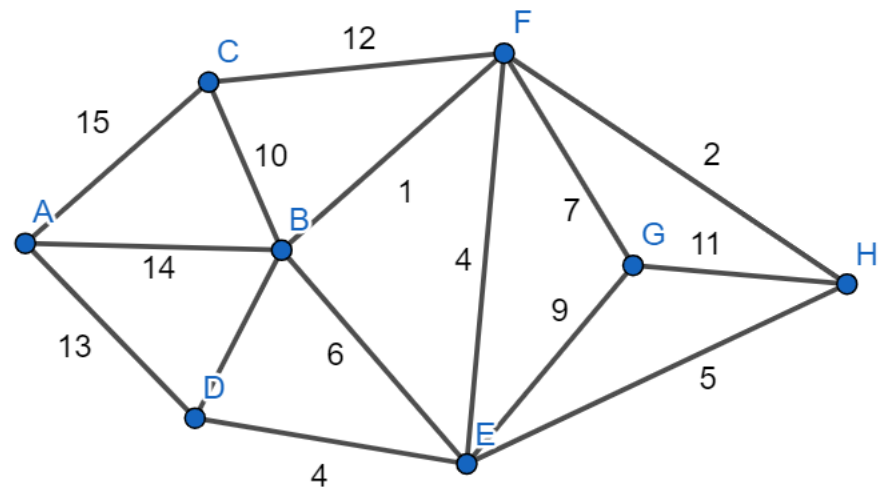


Bobotnya sama yaitu = 36

# SOAL LATIHAN

Tentukan dan gambarkan pohon merentang minimum dari graf pada gambar tersebut menggunakan :

- a) Algoritma Prim
- b) Algoritma Kruskal



# REFRENSI

Munir, Rinaldi, “Matematika Diskrit Ed. Revisi Ke-3”, Informatika Bandung, 2012