TERAPAN POHON BINER

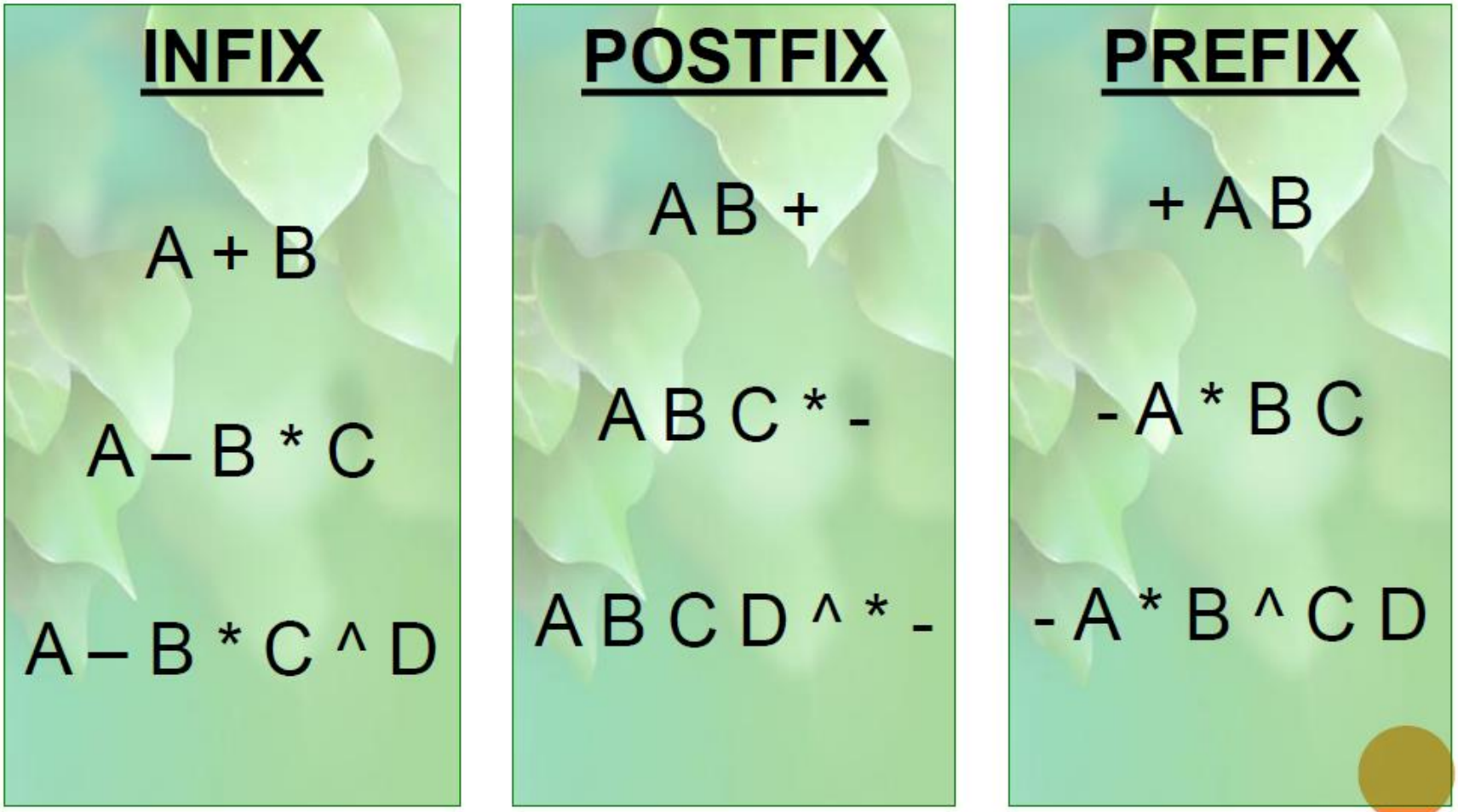
Terapan pohon biner di dalam ilmu komputer sangat banyak, diantaranya :

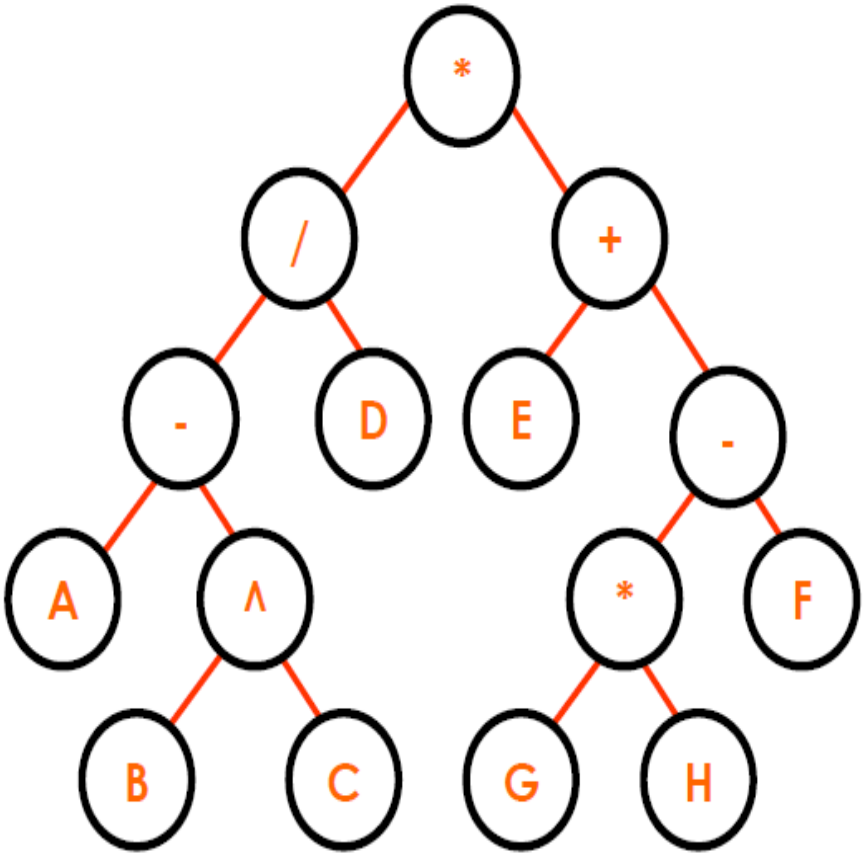
1. Pohon ekspresi
2. Pohon keputusan
3. Kode Prefiks
4. Kode Huffman
5. Pohon pencarian biner

Pohon Ekspresi

- Pohon ekspresi ialah pohon biner dengan daun berupa operand dan simpul dalam juga akar berupa operator.
- Tanda kurung tidak diperlukan bila suatu ekspresi aritmetik direpresentasikan sebagai pohon biner.
- Digunakan oleh compiler bahasa tingkat tinggi untuk mengevaluasi ekspresi yang ditulis dalam notasi infix, postfix dan prefix.

- INFIX : operator diantara 2 operand
- PREFIX : operator mendahului 2 operand
- POSTFIX : kedua operand mendahului operator


INFIX
A + B
A – B * C
A – B * C ^ D
POSTFIX
A B +
A B C * -
A B C D ^ * -
PREFIX
+ A B
- A * B C
- A * B ^ C D



Notasi Matematika :
(A–(B ^ C)) / D * (E + (G * H - F))

Infix :
A – B ^ C / D * E + G * H - F

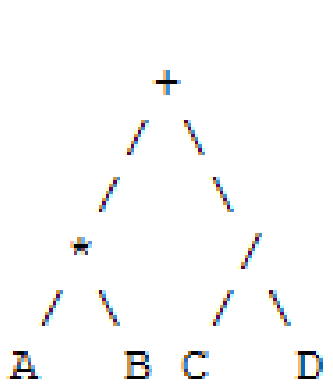
Prefix :
*** / - A ^ B C D + E - * G H F**

Postfix :
A B C ^ - D / E G H * F - + *

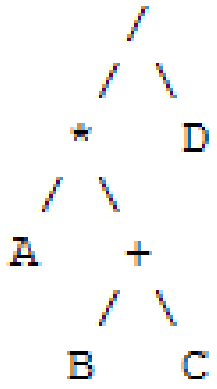
Infix	Postfix	Prefix
((A * B) + (C / D))	((A B *) (C D /) +)	(+ (* A B) (/ C D))
((A * (B + C)) / D)	((A (B C +) *) D /)	(/ (* A (+ B C)) D)
(A * (B + (C / D)))	(A (B (C D /) +) *)	(* A (+ B (/ C D)))

You can convert directly between these bracketed forms simply by moving the operato
superfluous brackets.

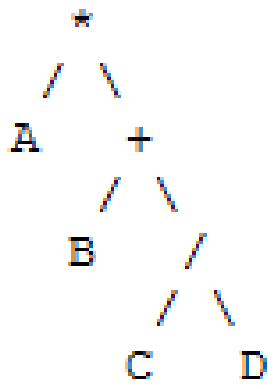
You can use a similar trick to convert to and from parse trees - each bracketed triplet



((A*B) + (C/D))



((A* (B+C)) /D)



(A* (B+ (C/D)))

LATIHAN : (NOTASI MATEMATIKA)

- Ubahlah notasi matematika berikut ini ke dalam bentuk pohon biner :
  - (A - B) * C / D
  - P * Q / R + (S – (T ^ U))
- Tentukan notasi infix, prefix dan postfix berdasarkan pohon biner yang telah dibuat

Notasi Postfix ke bentuk pohon

- Ubahlah notasi postfix berikut ini ke dalam bentuk pohon biner:
  (1) A B – C * D E S ^ / +
  (2) A B C E F * D + / - * S G + +

Pohon Keputusan

- Pohon keputusan digunakan untuk memodelkan persoalan yang terdiri dari serangkaian keputusan yang mengarah ke solusi.
- Tiap simpul dalam menyatakan keputusan, sedangkan daun menyatakan solusi.

1. Pohon Merentang

- Setiap graf terhubung mempunyai paling sedikit satu buah pohon merentang.
- Graf yang tidak mengandung sirkuit adalah pohon merentang itu sendiri.
- Pada graf yang mempunyai sirkuit, pohon merentangnya diperoleh dengan cara memutuskan sirkuit yang ada.

Pohon Merentang Minimum (Minimum spanning tree)

- Di antara semua pohon merentang di G, pohon merentang yang berbobot minimum dinamakan **pohon merentang minimum**
- **Terdapat 2 buah algoritma membangun pohon merentang minimum, yaitu :**

  Algoritma Prim.

  Algoritma Kruskal.

1. Algoritma Prim

- Algoritma Prim membentuk pohon merentang minimum langkah per langkah.
- Pada setiap langkah diambil sisi dari graf G yang mempunyai bobot minimum namun terhubung dengan pohon merentang minimum T yang telah terbentuk.

Langkah-langkah Algoritma Prim

1. Ambil sisi dari graf G yang berbobot minimum, masukkan ke dalam T.
2. Pilih sisi (u, v), yang mempunyai bobot minimum dan bersisian dengan simpul di T, tetapi (u, v) tidak membentuk sirkuit di T. Tambahkan (u, v) ke dalam T.
3. Ulangi langkah ke 2 sebanyak n – 2 kali.

Jumlah langkah seluruhnya di dalam Algoritma Prim adalah : 1 + (n – 2) = n – 1, yaitu sebanyak jumlah sisi di dalam pohon merentang dengan n buah simpul.

2. Algoritma Kruskal

- Pada Algoritma Kruskal, sisi-sisi graf diurutkan terlebih dahulu berdasarkan bobotnya dari kecil ke besar.

  Perbedaan prinsip antara algoritma Prim dan Kruskal adalah :

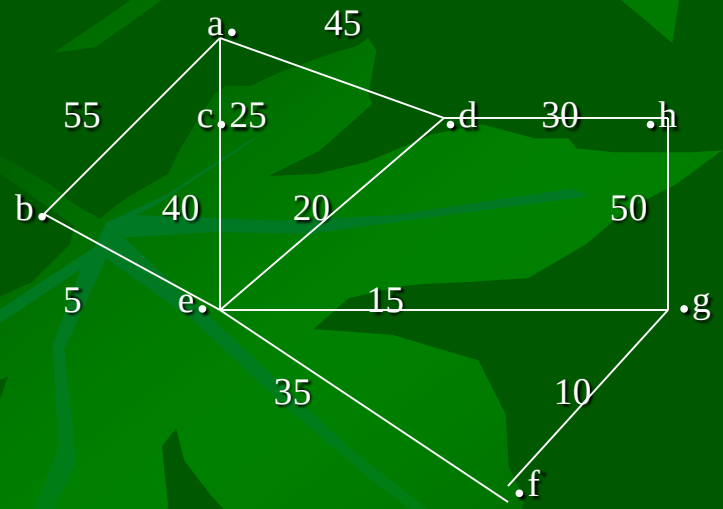
- Jika pada algoritma Prim, sisi yang dimasukkan ke dalam T harus bersisian dengan sebuah simpul di T, maka pada algoritma Kruskal sisi yang dipilih tidak perlu bersisian dengan sebuah simpul di T asalkan penambahan sisi tersebut tidak membentuk sirkuit.

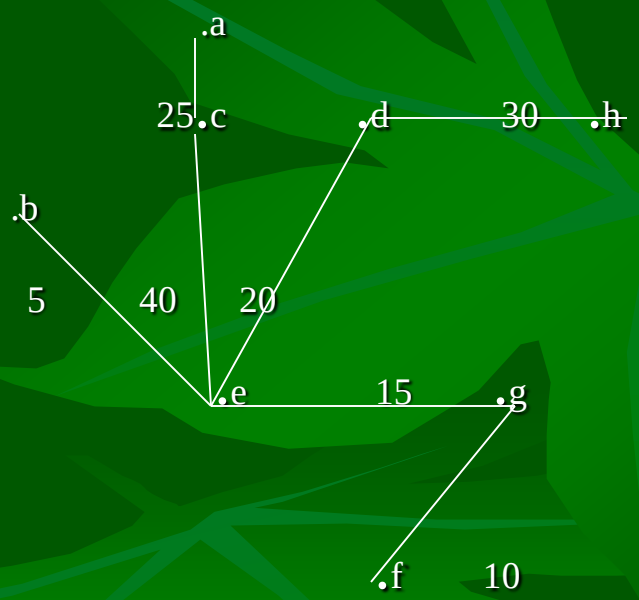
Langkah-langkah Algoritma Kruskal

Sisi-sisi dari graf diurutkan menaik berdasarkan bobotnya, dari bobot kecil ke bobot besar.

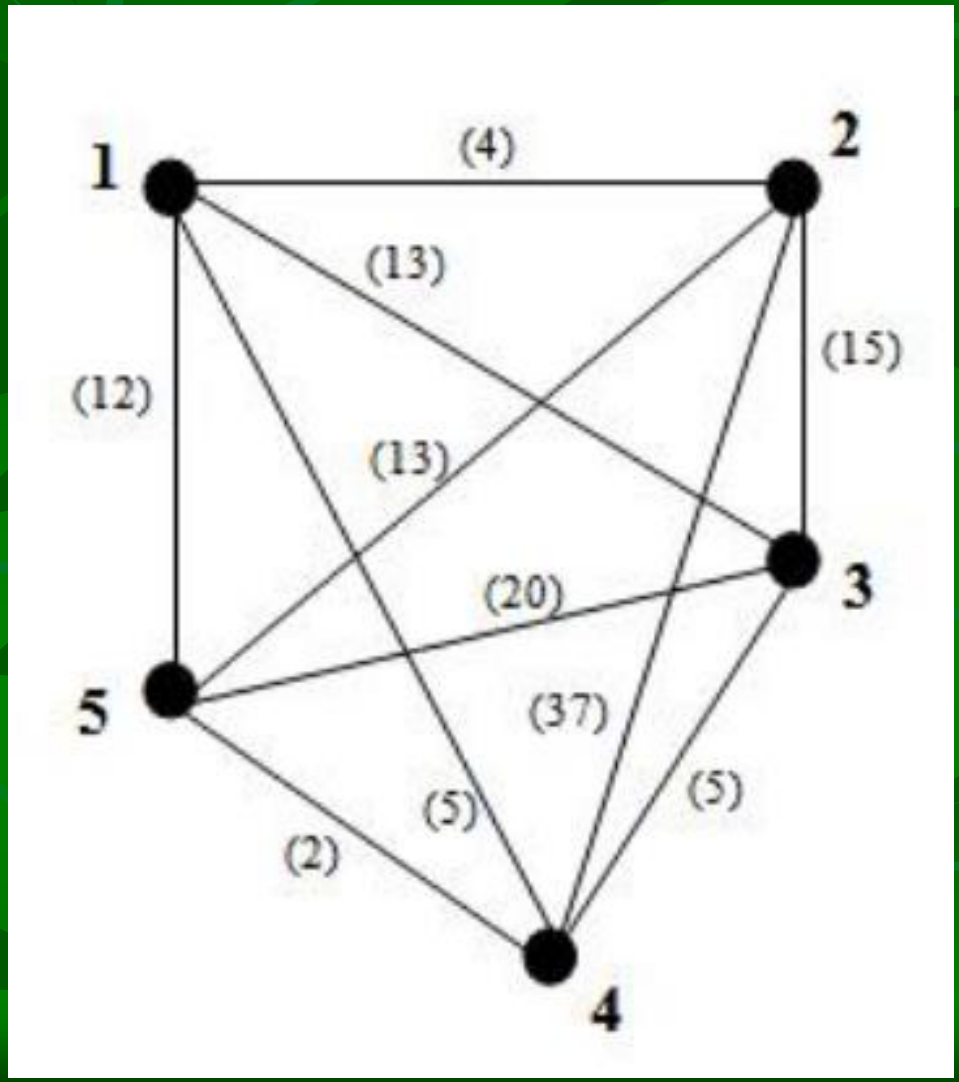
1. T masih kosong.
2. Pilih sisi (u, v) dengan bobot minimum yang tidak membentuk sirkuit di T. Tambahkan (u, v) ke dalam T.
3. Ulangi langkah ke 2 sebanyak n – 1 kali.

Contohnya: Pemerintah akan membangun jalur rel kereta api yang menghubungkan sejumlah kota. Karena biayanya mahal, pembangunan jalur ini tidak perlu menghubungkan langsung dua buah kota, tetapi cukup membangun jalur kereta seperti pohon rentang. Karena dalam sebuah graf mungkin saja terdapat lebih dari satu pohon rentang, maka harus dicari pohon rentang yang mempunyai jumlah jarak terpendek, dengan kata lain harus dicari pohon rentang minimum.



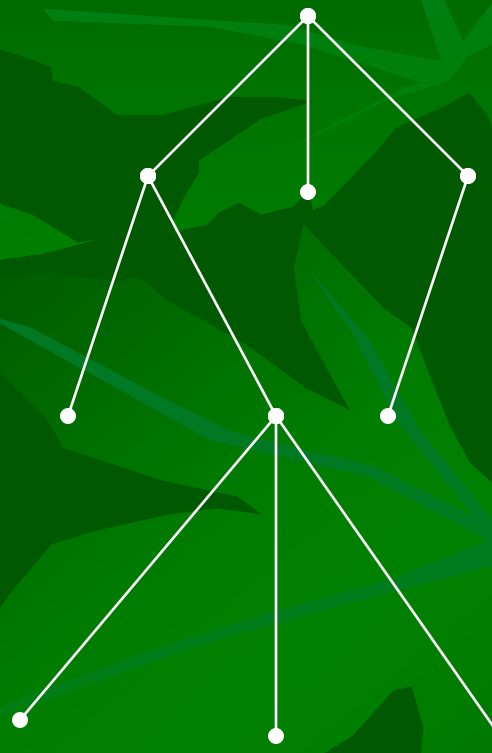


Soal :
Tentukan rentang pohon minimal graf berikut :



2. Pohon Berakar

- **Definisi : Pohon yang sebuah simpulnya diperlakukan sebagai akar dan sisi-sisinya diberi arah sehingga menjadi graf berarah.**
- **Akar mempunyai derajat masuk sama dengan nol dan simpul-simpul lainnya berderajat masuk sama dengan satu.**



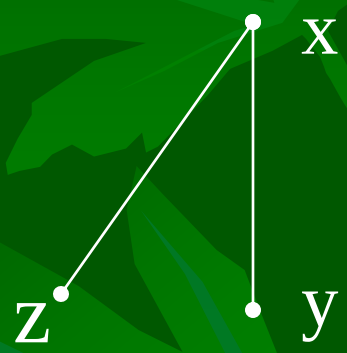
- Simpul yang mempunyai derajat keluar sama dengan nol disebut daun atau simpul terminal.
- Simpul yang mempunyai derajat keluar tidak sama dengan nol disebut simpul dalam atau simpul cabang.
- Setiap simpul di pohon dapat dapat dicapai dari akar dengan sebuah lintasan tunggal.

- Sembarang pohon tak berakar dapat diubah menjadi pohon berakar dengan memilih sebuah simpul sebagai akar.
- Pemilihan simpul yang berbeda akan menghasilkan pohon berakar yang berbeda.
- Arah sisi di dalam pohon dapat dibuang, karena setiap simpul di pohon harus dicapai dari akar, maka lintasan di dalam pohon berakar selalu dari atas ke bawah.

6. Terminologi pada Pohon Berakar

- Anak dan Orang tua.

  Misalkan x adalah simpul di dalam pohon berakar, simpul y dikatakan anak simpul x jika ada sisi dari simpul x ke simpul y dan simpul x disebut orang tua simpul y.



Lintasan (path)

- Lintasan dari simpul v_1 ke simpul v_k adalah runtunan simpul-simpul $v_1, v_2, v_3, \ldots, v_k$ sedemikian sehingga v_i adalah orangtua dari v_{i+1} untuk $1 \leq i \leq k$.
- Panjang lintasan adalah jumlah sisi yang dilalui dalam suatu lintasan, yaitu $k - 1$.

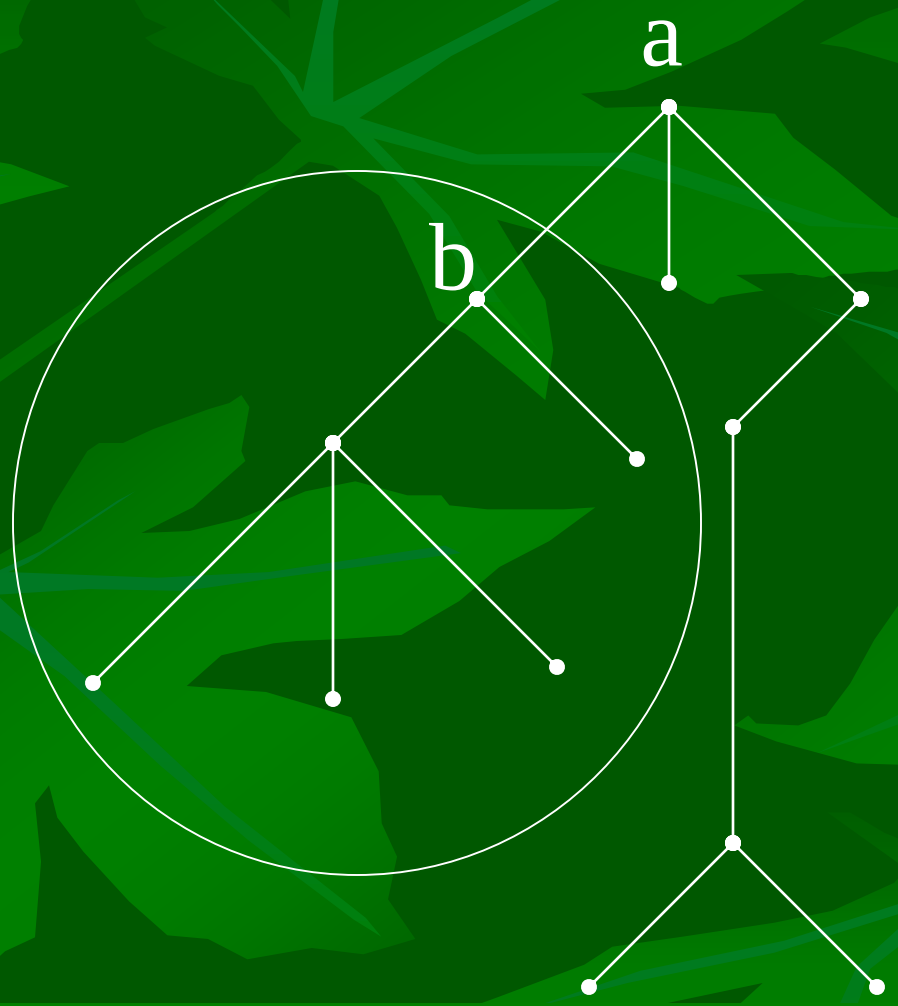
Keturunan dan Leluhur

- Jika terdapat lintasan dari simpul x ke simpul y di dalam pohon, maka x adalah leluhur dari simpul y, dan y adalah keturunan simpul x.

Saudara Kandung

- Simpul yang berorangtua sama adalah saudara kandung satu sama lain.

Upapohon (Subtree)



Pohon T dengan upapohon T’ pada bagian yang dilingkari.

Pohon T dengan akar a dan upapohon T’ dengan akar b.

Derajat (degree)

- Derajat sebuah simpul pada pohon berakar adalah jumlah upapohon atau jumlah anak pada simpul tersebut
- Derajat maksimum dari semua simpul merupakan derajat pohon itu sendiri.

Aras (level) atau Tingkat

- Akar mempunyai aras 0, sedangkan aras simpul lainnya = 1 + panjang lintasan dari akar ke simpul tersebut.

Tinggi (height) atau Kedalaman (depth)

- Aras maksimum dari suatu pohon disebut tinggi atau kedalaman, atau tinggi pohon adalah panjang maksimum lintasan dari akar ke daun.

Pohon Terurut

- Pohon berakar yang urutan anak-anaknya penting disebut pohon terurut (ordered tree)
- Pada pohon terurut, urutan anak-anak dari simpul dalam dispesifikasikan dari kiri ke kanan.

Pohon n-ary

- Pohon berakar yang setiap simpul cabangnya mempunyai paling banyak n buah anak disebut pohon n-ary.
- Pohon n-ary dikatakan teratur atau penuh jika setiap simpul cabangnya mempunyai tepat m buah anak.