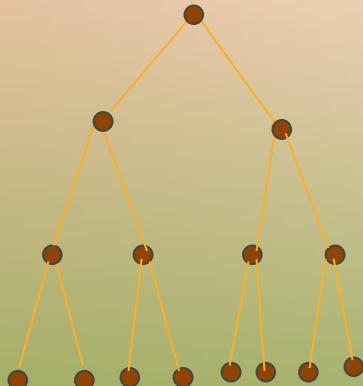


PERTEMUAN 14

- POHON BINER
- POHON EKSPRESI
- POHON KEPUTUSAN

POHON BINER

- Definisi pohon yang setiap simpul cabangnya mempunyai paling banyak 2 buah child (anak)
- Left subtree (sub pohon kiri) adalah Pohon yang akarnya adalah left child (anak kiri)
- Right subtree (sub pohon kanan) adalah Pohon yang akarnya adalah right child (anak kanan)
- Skewed tree (pohon condong) adalah : Pohon yang semua simpulnya terletak di bagian kiri saja atau bagian kanan saja
- Skew left (pohon condong kiri) adalah : Pohon yang condong ke kiri
- Skew right (pohon condong kanan) adalah : Pohon yang condong ke kanan
- Full binary tree (pohon biner penuh) adalah : Pohon biner yang setiap simpulnya mempunyai tepat 2 buah child (anak), kiri dan kanan, kecuali simpul pada level bawah
- Pohon biner penuh dengan tinggi h memiliki jumlah daun sebanyak 2^h , dan jumlah seluruh simpul adalah :

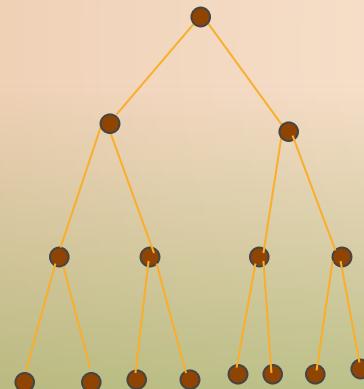


$$S = 2^0 + 2^1 + 2^2 + \dots + 2^h = 2^{h+1} - 1$$

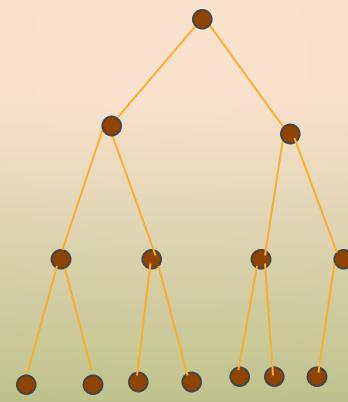
Pohon biner penuh

POHON BINER SEIMBANG

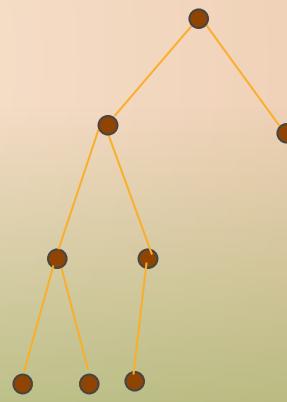
- Definisi pohon biner yang perbedaan tinggi antara subpohon kiri dan subpohon kanan maksimal 1
- Pada pohon biner seimbang dengan tinggi h , semua daun berada pada level h atau $h - 1$
- Untuk membuat pohon seimbang, tinggi pohon secara keseluruhan harus dibuat seminimal mungkin
- Untuk memperoleh tinggi minimum, setiap level harus mengandung jumlah simpul sebanyak mungkin



Pohon biner seimbang



Pohon biner seimbang

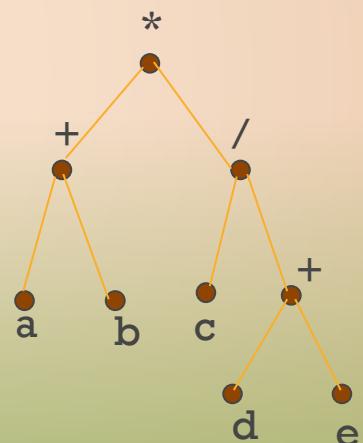


Bukan Pohon biner seimbang

POHON EKSPRESI

- Definisi pohon biner dengan daun menyatakan operand dan simpul dalam (termasuk akar) menyatakan operator.
- Tanda kurung tidak lagi diperlukan bila suatu ekspresi aritmetik direpresentasikan sebagai pohon biner
- Pohon ekspresi digunakan oleh compiler bahasa tingkat tinggi untuk mengevaluasi ekspresi yang ditulis dalam notasi :
 - Infix
 - Operator berada di antara 2 buah operand
 - Prefix (polish notation)
 - Operator mendahului 2 buah operand-nya
 - Postfix (inverse polish notation)
 - Kedua operand mendahului operatornya
- Contoh :

$(a + b)^*(c/(d + e))$	→ infix
$* + a b / c + d e$	→ prefix
$a b + c d e + / *$	→ postfix

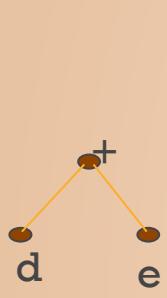


Pohon ekspresi dari
 $(a + b)^*(c/(d + e))$

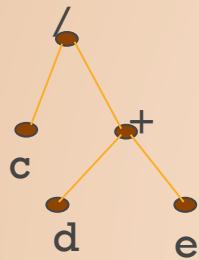
Penjelasan
slide
berikutnya

LANJUTAN PENJELASAN

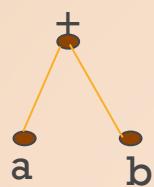
Pembentukan pohon ekspresi $(a + b)^*(c/(d + e))$



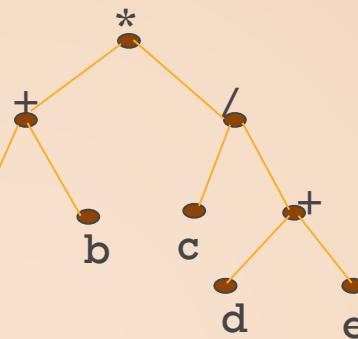
(i)



(ii)



(iii)

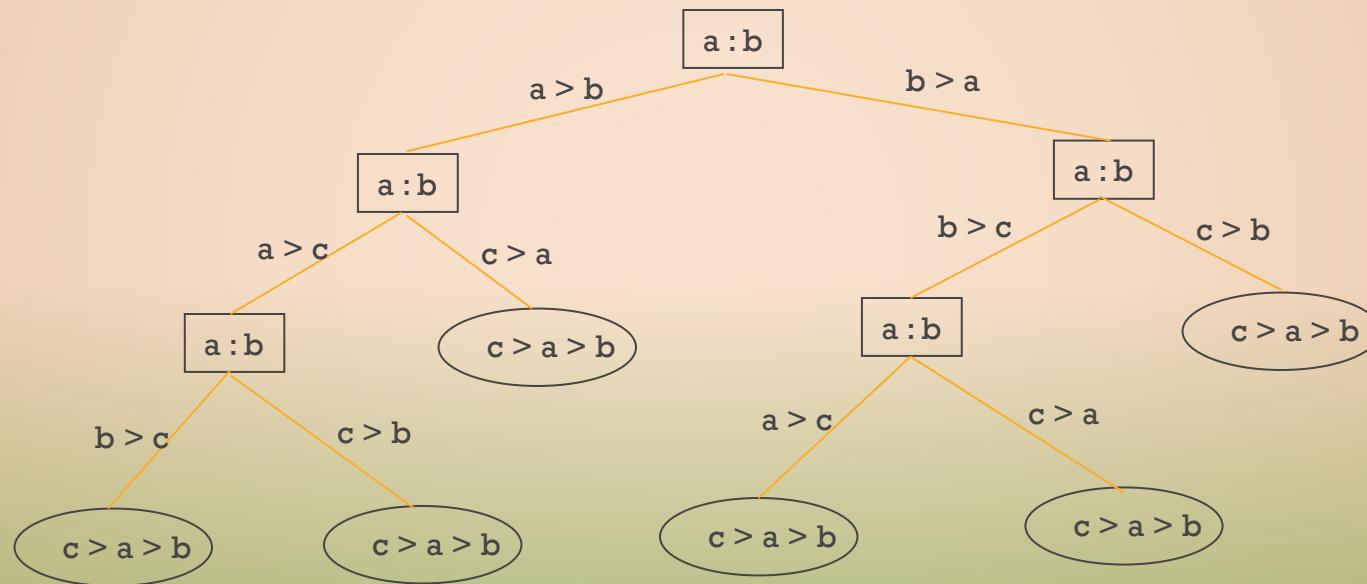


(iv)

- Urutan prioritas penggerjaan operator :
 1. Perkalian (*) dan pembagian (/) → lebih tinggi
 2. Penjumlahan (+) dan pengurangan (-)

POHON KEPUTUSAN

- Digunakan untuk memodelkan persoalan yang terdiri dari serangkaian keputusan yang mengarah ke solusi
- Tiap simpul dalam menyatakan keputusan
- Daun menyatakan solusi



Pohon keputusan untuk mengurutkan 3 buah elemen

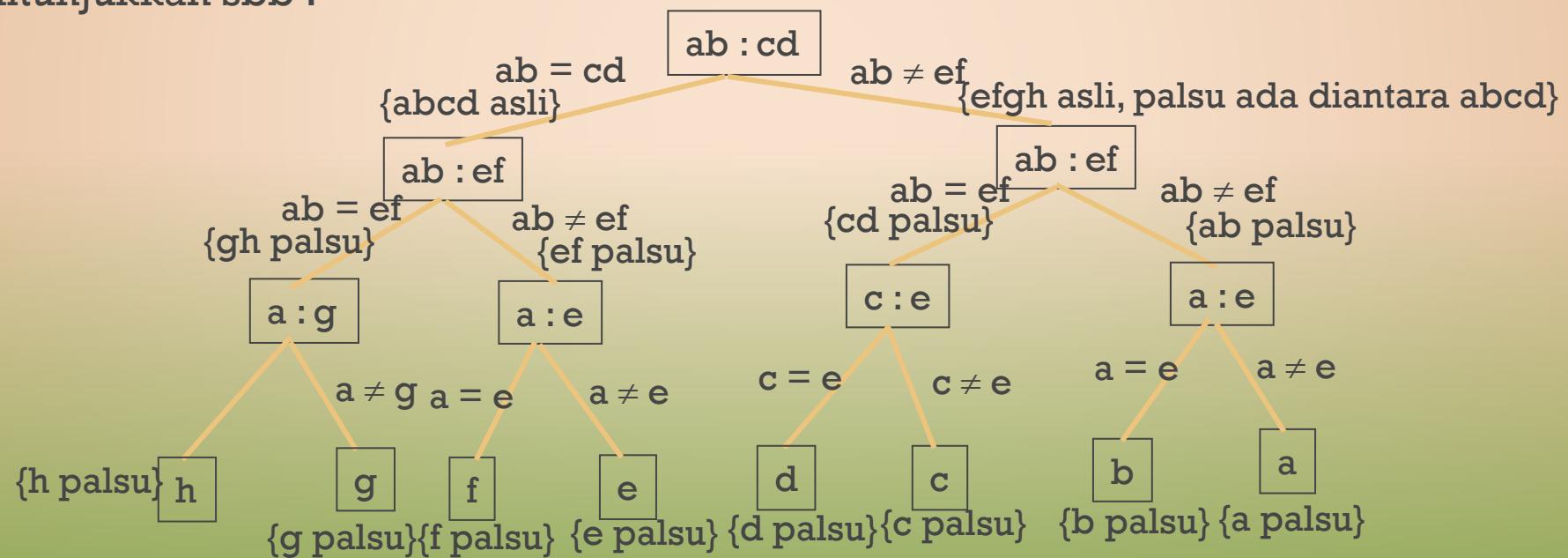
POHON KEPUTUSAN

Contoh :

Diketahui 8 buah koin uang logam. Satu dari delapan koin ternyata palsu. Koin yang palsu mungkin lebih ringan atau lebih berat daripada koin yang palsu. Misalkan tersedia sebuah timbangan neraca yang sangat teliti. Buatlah pohon keputusan untuk mencari uang palsu dengan cara menimbang paling banyak hanya 3 kali saja

Penyelesaian :

Misalkan 8 koin itu dinamai a,b,c,d,e,f,g,h. Daun menyatakan koin yang palsu. Pohon keputusan untuk mencari koin yang palsu ditunjukkan sbb :



1. Rinaldi Munir. (2016). Matematika Diskrit. Bandung : Penerbit Informatika
2. Diktat dan Handout Matematika Diskrit. Tim Dosen Universitas Indraprasta PGRI .