Tata Bahasa Pemrograman

- Bahasa dapat digambarkan sebagai suatu tatatanan yang membentuk hirarki
- Bahasa yang menggambarkan suatu urutan yang tertata secara valid disebut dengan **formal languange**
- Untuk membentuk formal languange diperlukan :
  - set letter yang diterima oleh languange (‘L’)
  - konsep pembentukan ‘sequence’ dari item-item yang ada

Terminal Symbols, Alphabet dan Strings

- **Alphabet** : representasi letter dari L yang ditulis dalam huruf kecil (a,b,c,...z)
- **Terminal Simbol** (T) : individual karakter, termasuk di dalamnya adalah alphabet

  {a,b,....z, 0,1,...9}
- **Strings** : urutan simbol terbatas (sequence finite symbol)
  - contoh : aba, ab12z, axy
  - operasi yang dapat dilakukan pada strings : substrings, concatenation

Non Terminal

- Non Terminal (NT) : kumpulan string alphabeth yang ada pada formal language
- Word atau vocabulary unit dalam L yang mencerminkan kategori sintax (noun, verb, dll) merupakan NT.
- Non Terminal direpresentasikan dalam huruf besar (A,B,….) dan diapit dengan tanda <..> untuk membedakannya dengan string

Produksi

- Konsep T dan NT direpresentasikan dalam Produksi
- Bentuk Umum Produksi

  NT ::= string T atau NT
- Contoh :

  <Noun Phrase> ::= <Article><Noun>

  <Article> ::= a | an | the

  <Noun> ::= boy | apple

Komponen Tata Bahasa

- Set dari simbol Terminal (T)
- Set dari simbol Non Terminal (NT)
- Set dari Produksi
- Distinguished symbol dalam grammar

Derivasi, Reduksi dan Pohon Sintaks

- **Derivasi** merupakan uraian dari suatu Produksi
  - contoh : <Noun Phrase> ⇒ <Article> <Noun>
    <Article><Noun> ⇒ the <Noun>
    the <Noun> ⇒ the boy
- **Reduksi** proses perubahan string ke dalam NT sehingga menjadi suatu grammar production
  - contoh : <Sentence> ::= <Noun Phrase><Verb Phrase>
    <Noun Phrase>::= <Article><Noun>
    <Verb Phrase> ::= <Verb><Noun>
    <Article> ::= a | an | the
    <Noun> ::= boy | apple
    <Verb> ::= ate

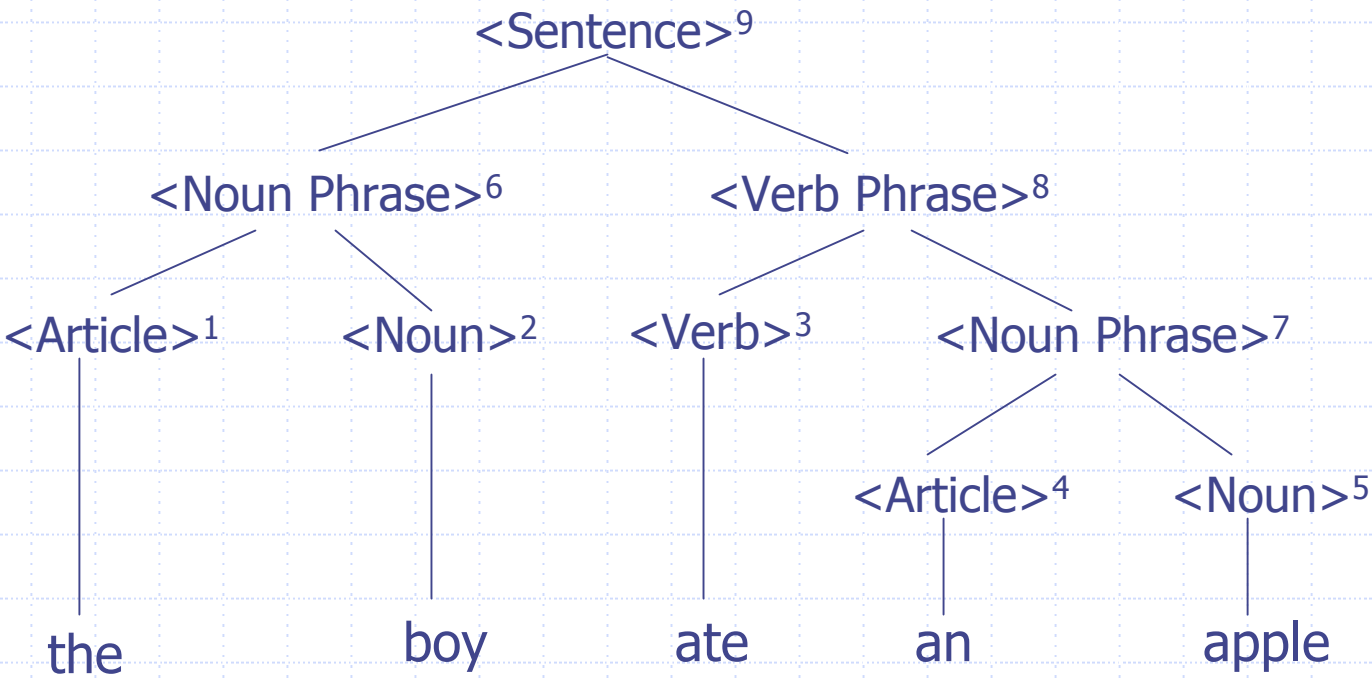
Derivasi, Reduksi dan Pohon Sintaks (cont.)

- Reduksinya menjadi :

Step	String
0	the boy ate an apple
1	<Article> boy ate an apple
2	<Article><Noun>ate an apple
3	<Article><Noun><Verb>an apple
4	<Article><Noun><Verb><Article>apple
5	<Article><Noun><Verb><Article><Noun>
6	<NounPhrase><Verb><Article><Noun>
7	<NounPhrase><Verb><Noun Phrase>
8	<NounPhrase><Verb Phrase>
9	<Sentence>

Derivasi, Reduksi dan Pohon Sintaks (cont.)

- Reduksi dapat direpresentasikan dalam **Pohon Sintaks**



Derivasi, Reduksi dan Pohon Sintaks (cont.)

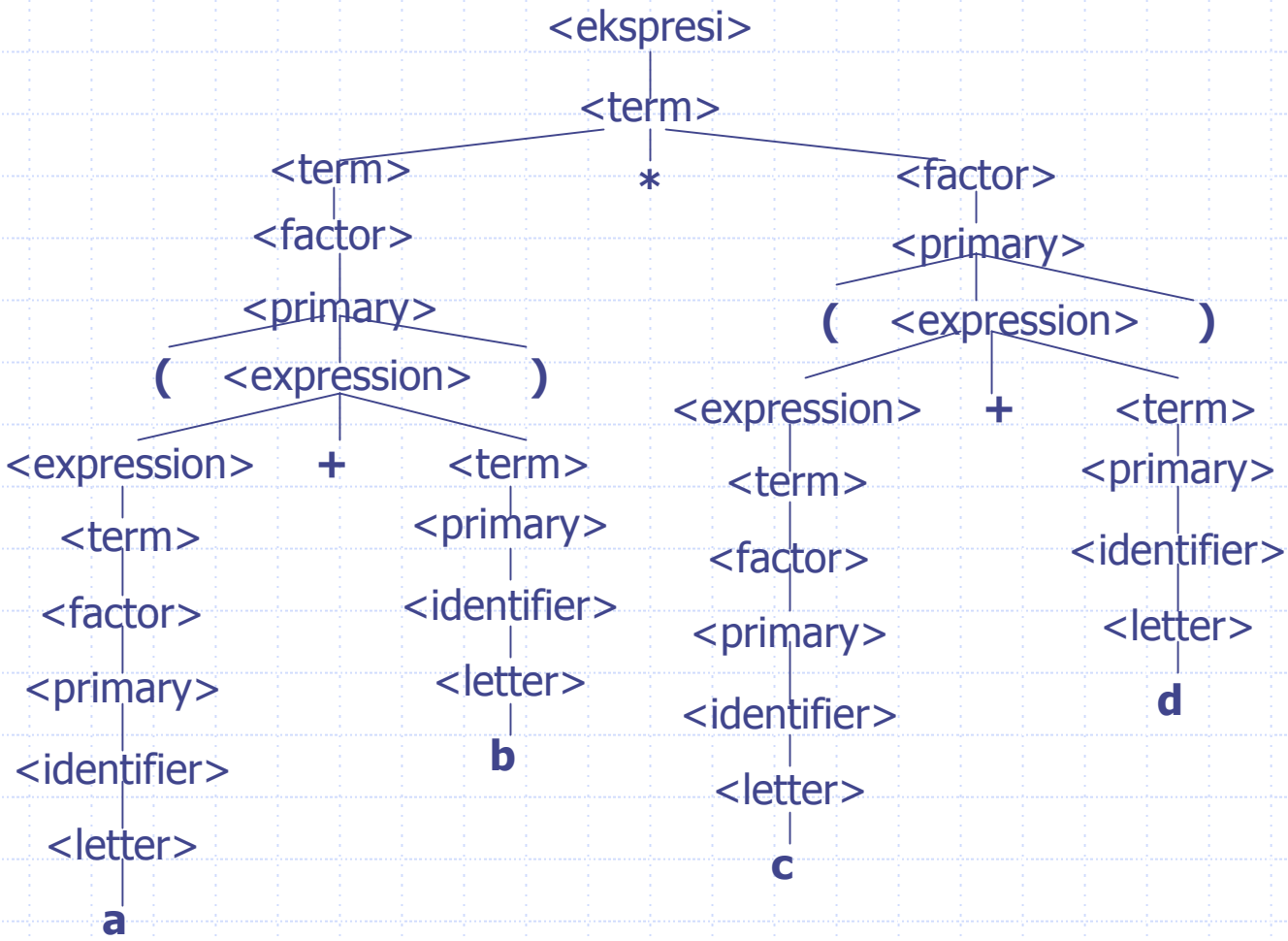
- Contoh grammar dalam ekspressi aritmetika dalam notasi Backus Naur Form (BNF) :

<expression>	::= <expression> + <term> | <term>
<term>	::= <term> * <factor> | <factor>
<factor>	::= <factor> ↑ <primary> | <primary>
<primary>	::= <identifier> | <constant> | (<ekspression>)
<identifier>	::= <letter> | <identifier> [<letter> | <digit>]
<constant>	::= [{ + | - }] <digit> | <constant><digit>
<letter>	::= a | b | c | ……. | z
<digit>	::= 0 | 1| 2| …..| 9

- Produksi dapat memiliki sifat Recursive, yaitu bila ada NT yang memanggil dirinya sendiri

Derivasi, Reduksi dan Pohon Sintaks (cont.)

Buatlah syntax tree untuk ekpresi **(a + b) * (c + d)** !



Klasifikasi Grammar

Chomsky (1963) mengklasifikasikan Grammar menjadi 4 kategori :

- **Grammar Type-0**, disebut **Phrase Structure Grammar** atau grammar tidak terbatas

  Bentuknya : $\alpha ::= \beta$

  Dimana, α dan β dapat berupa T atau NT, yang dapat saling bersubstitusi , karenanya tidak relevan untuk bahasa pemrograman

- **Grammar Type-1**, disebut **Context Sensitive Grammar (CSG)**, produksi dari grammar ini adalah derivasi/reduksi dari particular string yang hanya terdapat pada particular context

  Bentuknya : $\alpha \mathbf{A} \beta ::= \alpha \pi \beta$

  Dimana, π menggantikan A untuk menutupi string α dan β. Grammar ini-pun tidak relevan untuk bahasa pemrograman

Klasifikasi Grammar (cont.)

- **Grammar Type-2**, disebut **Context Free Grammar (CFG)**, grammar ini tidak membutuhkan context pengenal atau derivasi

  Bentuknya : **A ::= π**

  Grammar ini digunakan pada ALGOL-60 dan PASCAL

- **Grammar Type-3**, disebut **Linier atau RegularGrammar**, dimana RHS dapat berupa single terminal simbol danterminal simbol atau kebalikannya

  Bentuknya : **A ::= t B | t atau A ::= B t | t**

  Bentuk ini banyak dijumpai pada bahasa pemrograman