# **Spesifikasi Sintaks Java dalam Extended Backus Normal Form (eBNF)**

# **Mata Kuliah IF5020 - Algoritma dan Pemrograman A**

Oleh :

1. Sukamto - 23518017

2. Andreas Novian Dwi T. - 23518002

Aturan:

Non-terminal : Tulisan yang tidak dicetak tebal.

Terminal : Tulisan yang dicetak tebal.

**Program Declaration**

1. <compilation unit> ::= <package declaration> {<import declaration>} {<class declaration>} | {<import declaration>} {<class declaration>}
2. <package declaration> ::= **package** <package name> **;**
3. <import declaration> ::= **import** <package name> <import ending>
4. <package name> ::= <identifier> {**.**<identifier>}
5. <import ending> ::= **. \* ;** | **;**

**Class**

1. <class declaration> ::= <class modifier> <class modifier extension>
2. <class modifier extension> ::= <class modifier2> **class** <class declaration extension> | <abstract modifier> <type declaration> | **interface** <interface declaration>
3. <type declaration> ::= **class** <abstract class declaration>| **interface** <interface declaration>
4. <class declaration extension> ::= <identifier> <super> <interfaces> <class body>
5. <abstract class declaration> ::= <identifier> <super> <interfaces> <abstract class body>
6. <class modifier> ::= **public** | 𝝴
7. <class modifier2> ::= **final** | 𝝴
8. <abstract modifier> ::= **abstract**
9. <super> ::= **extends** <identifier> | 𝝴
10. <interfaces> ::= **implements** <identifier> { **,** <identifier> } | 𝝴
11. <class body> ::= **{** {<class body declaration>} **}**
12. <class body declaration> ::= <abstract method modifier> <body declaration> | **private** <abstract body declaration>
13. <abstract method modifier> ::= **public** | **protected |** 𝝴
14. <abstract class body> ::= **{** {<abstract class body declaration>} **}**
15. <abstract class body declaration> ::= <abstract method modifier> <abstract body declaration> | **private** <abstract body declaration>
16. <body declaration> ::= <field declaration> | <method initializer> | **final** <final declaration> | **static** <static option> | <data type declaration>
17. <data type declaration> ::= <data primitive> <data primitive declaration> | <identifier> <declaration type>
18. <data primitive declaration> ::= <identifier> <variable or method option> | <array after data type> <variable looping> **;**
19. <declaration type > ::= <constructor declaration> | <throws> | <variable operator> <variable looping> **;** | <array after data type>
20. <abstract body declaration> ::= <body declaration> | <abstract method declaration>
21. <final declaration> ::= <field modifier3 declaration> | <final additional mod>
22. <static option> ::= <data type declaration> | <static declaration>
23. <static declaration> ::= <static method declaration> | <static initializer> | <synchronized modifier> <synchronized method declaration> | 𝝴
24. <static initializer> ::= <block> | <static field declaration>

**Interface**

1. <interface declaration> ::= <identifier> <extends interfaces> <interface body>
2. <extends interfaces> ::= **extends** <identifier> { **,** <identifier>} | 𝝴
3. <interface body> ::= **{** {<interface member>} **}**
4. <interface member> ::= <constant declaration> | <abstract method declaration>

**Constructor**

1. <constructor declaration> ::= <parameters> <throws> <constructor body>
2. <parameters> ::= **(** {<formal parameter>} **)**
3. <formal parameter> ::= <data type> <identifier> <comma option>
4. <comma option> ::= **,** | 𝝴
5. <throws> ::= **throws** <identifier> {**,**<identifier>} | 𝝴
6. <constructor body> ::= **{** {<explicit constructor invocation>} {<block statement>} **}**
7. <explicit constructor invocation>::= **this (** <argument list> **) ;** | **super (** <argument list> **) ;**

**Field**

1. <field declaration> ::= <field additional modifiers> <data type> <variable declarators> **;**
2. <static field declaration> ::= <field modifier2 initializer> <data type> <variable declarators> **;**
3. <field modifier2> ::= **transient**
4. <field modifier3> ::= **volatile**
5. <static modifier> ::= **static**
6. <field additional modifiers> ::= <field modifier2> <static modifier initializer> | <field modifier3> <field modifier3 declaration>
7. <static modifier initializer> ::= <field modifier3> <static modifier> | <static modifier> <field modifier3 initializer> | 𝝴
8. <field modifier2 initializer> ::= <field modifier3> <field modifier2 option> | <field modifier2> <field modifier3 initializer>
9. <field modifier2 option> ::= **transient** | 𝝴
10. <field modifier3 initializer> ::= <field modifier3> | 𝝴
11. <field modifier3 declaration> ::= <field modifier2> <static modifier>
12. <variable or method option> ::= <parameters> <throws> <block> | <variable option> <variable looping> **;**
13. <variable declarators> ::= <variable declarator> { **,** <variable declarator>}
14. <variable declarator> ::= <identifier> <variable option> | <array after data type>
15. <array after data type> ::= **[ ]** { **[ ]** } <array declaration>
16. <array declaration> ::= <identifier><array initializer> | 𝝴
17. <variable option> ::= <variable operator> | **[ ]** { **[ ]** }<array initializer>
18. <variable option2> ::= <variable initializer> | **{** <array type initializer **}**
19. <variable operator> ::= **=** <variable initializer> | 𝝴
20. <variable initializers> ::= <variable initializer> { **,** <variable initializer> } | 𝝴
21. <variable initializer> ::= <expression>
22. <variable looping> ::= { **,** <identifier> <variable option> }
23. <array initializer> ::= **= {** <array type initializer> **}**
24. <array type initializer> ::= <variable initializers> | **{** <variable initializers> **}** { **, {** <variable initializers> **}** }
25. <constant declaration> ::= <constant modifiers> <data type> <variable declarator> **;**
26. <constant modifiers> ::= **public** | **static** | **final**

**Method**

1. <method initializer> ::= <method declaration> <block> | <native method declaration>
2. <method declaration> ::= <method additional modifier> **void** <method declarator> <throws> | **void** <method declarator> <throws>
3. <static method declaration> ::= <static additional mod> <result type> <method declarator> <throws> | **void** <method declarator> <throws>
4. <result type> ::= <data type> | **void**
5. <final modifier> ::= **final**
6. <synchronized modifier> ::= **synchronized**
7. <method additional modifier> ::= <synchronized modifier> <synchronized additional mod>
8. <static additional mod> ::= <final modifier> <synchronized mod initializer>
9. <synchronized method declaration> ::= <final mod initializer> | 𝝴
10. <final additional mod> ::= <synchronized modifier> <static mod initializer> | 𝝴
11. <synchronized additional mod> ::= <static modifier> <final mod initializer> | <final modifier> <static mod initializer> | 𝝴
12. <static mod initializer> ::= <static modifier> | 𝝴
13. <final mod initializer> ::= < final modifier > | 𝝴
14. <synchronized mod initializer> ::= <synchronized modifier> | 𝝴
15. <method declarator> ::= <identifier> <parameters>
16. <native method declaration> ::= <native modifier> <result type> <method declarator> <throws>
17. <native modifier> ::= **native**
18. <abstract method declaration> ::= <abstract modifier> <result type> <method declarator> <throws>
19. <method invocation> ::= <identifier> **(** <argument list> **)**

**Data Type**

1. <data type> ::= <data primitive> | <identifier> { **[ ]** }
2. <data primitive> ::= <primitive type> { **[ ]** }
3. <primitive type> ::= **boolean | float** | **double | byte** | **short** | **int** | **long** | **char**

**Statement**

1. <block> ::= **{** {<block statement>} **}**
2. <block statement> ::= <data primitive> <variable declarators> **;** | <identifier><block statement option> | <statement without expression statement> | <expression statement without identifier>
3. <block statement option> ::= { **[ ]** } <variable declarators> **;** | **:** <statement> | <post identifier> **;**
4. <statement without expression statement> ::= <block> | <empty statement> | <switch statement> | <do statement> | <break statement> | <continue statement> | <return statement> | <synchronized statement> | <throws statement> | <try statement> | <if statement> | <while statement> | <for statement>
5. <statement> ::= <statement without expression statement> | <expression statement>
6. <empty statement> ::= **;**
7. <labeled statement> ::= <identifier> **:** <statement>
8. <expression statement without identifier> ::= <statement expr> **;**
9. <expression statement> ::= <statement expr> **;** | <identifier> <identifier statement>
10. <identifier statement> ::= <post identifier> **;** | **:** <statement>
11. <statement expression> ::= <statement expr> | <identifier> <post identifier>
12. <statement expr> ::= **super** <this super option> | **this** <this super option> | <increment decrement> <identifier> | **new** <identifier> <class instance creation expression>
13. <this super option> ::= **.** <identifier> {**.** <identifier>} <array access> <assignment> | <parameters>
14. <post identifier> ::= <increment decrement> | **(** <argument list> **)** | { **.** <identifier> } <assignment>

**Branching**

1. <if statement>::= **if (** <expression> **)** <statement> <else statement>
2. <else statement> ::= **else** <statement> | 𝝴
3. <switch statement> ::= **switch (** <expression> **)** <switch block>
4. <switch block> ::= **{** {<switch block statement group>} **}**
5. <switch block statement group> ::= <switch label> {<block statement>}
6. <switch label> ::= **case** <expression> **:** | **default :**

**Looping**

1. <while statement> ::= **while (** <expression> **)** <statement>
2. <do statement> ::= **do** <statement> **while (** <expression> **) ;**
3. <for statement> ::= **for (** <for init> **;** <expression> **;** <for update> **)** <statement>
4. <local variable declaration> ::= <data type> <variable declarators> | <variable declarators>
5. <for init> ::= <statement expression list> | <local variable declaration> | 𝝴
6. <for update> ::= <statement expression list> | 𝝴
7. <statement expression list> ::= <statement expression> { **,**<statement expression> }
8. <break statement> ::= **break** <break continue identifier> **;**
9. <continue statement> ::= **continue** <break continue identifier> **;**
10. <break continue identifier> ::= <identifier> | 𝝴
11. <return statement> ::= **return** <expression> **;**
12. <throws statement> ::= **throw** <expression> **;**
13. <synchronized statement> ::= **synchronized (** <expression> **)** <block>
14. <try statement> ::= **try** <block> <catch statement>
15. <catch statement> ::= <catches> <finally initializer> | <finally statement>
16. <finally initializer> ::= <finally statement> | 𝝴
17. <catches> ::= <catch clause> {<catch clause>}
18. <catch clause> ::= **catch (** <formal parameter> **)** <block>
19. <finally statement> ::= **finally** <block>

**Expression**

1. <expression> ::= <additive expression> <multiplicative operator>
2. <multiplicative operator> ::= **\*** <expression>| **/** < expression>| **%** <expression> | 𝝴
3. <additive expression> ::= <shift expression> <additive operator>
4. <additive operator> ::= **+** <additive expression> | **-** <additive expression> | 𝝴
5. <shift expression> ::= <relational expression> <shift operator>
6. <shift operator> ::= **>>** <shift expression> | **<<** <shift expression> | **>>>** <shift expression> | 𝝴
7. <relational expression> ::= <equality expression> <relational operator>
8. <relational operator> ::= **<** <relational expression>| **>** <relational expression>| **<=** <relational expression>| **>=** <relational expression>| **instance of** <relational expression> | 𝝴
9. <equality expression> ::= <and expression> <equality operator>
10. <equality operator> ::= **==** <equality expression> | **!=** <equality expression> | 𝝴
11. <and expression> ::= <exclusive or> <and operator>
12. <and operator> ::= **&** <and expression> | 𝝴
13. <exclusive or> ::= <inclusive or> <exclusive operator>
14. <exclusive operator> ::= **^** <exclusive or> | 𝝴
15. <inclusive or> ::= <conditional and> <inclusive or operator>
16. <inclusive or operator> ::= **|** <inclusive or> | 𝝴
17. <conditional and> ::= <conditional or> <conditional and operator>
18. <conditional and operator> ::= **&&** <conditional and> | 𝝴
19. <conditional or> ::= <unary expression> <conditional or operator>
20. <conditional or operator> ::= **?** <expression> **:** <expression> | 𝝴
21. <cast type> ::= <primitive type> | <identifier>
22. <unary expression> ::= <expression name> <post expression name> | <increment decrement> <expression name> | <primary> | **(** <cast type> **)** <unary expression> | **+** <unary expression> | **-** <unary expression> | **~** <unary expression>
23. <increment decrement> ::= **--** | **++**
24. <post expression name> ::= <increment decrement> | {**.** <identifier>} <array or assignment> | 𝝴
25. <array or assignment> ::= <array access> | <assignment operator> <expression>
26. <primary> ::= <literal> { **.** <method invocation> } | **new** <instance creation expression> { **.** <method invocation> } | <field access> <assignment>
27. <instance creation expression> ::= <primitive type> <dim expressions> <dims> | <identifier> <creation expression>
28. <creation expression> ::= <class instance creation expression> | <array creation>
29. <class instance creation expression> ::= **(** <argument list> **)**
30. <argument list> ::= <expression> { **,** <expression> } | 𝝴
31. <array creation> ::= <dim expressions> <dims>
32. <dim expressions> ::= <dim expression> {<dim expression>}
33. <dim expression> ::= **[** <expression> **]**
34. <dims> ::= { **[ ]** }
35. <assignment> ::= <assignment operator> <expression> | 𝝴
36. <assignment operator> ::= **=** | **\*=** | **/=** | **%=** | **+=** | **-=** | **<<=** | **>>=** | **>>>=** | **&=** | **^=** | **|=**
37. <field access> ::= **super.** <identifier> {**.** <identifier>} <array access> | **this.** <identifier> {**.** <identifier>} <array access>
38. <array access> ::= { **[** <expression> **]** }
39. <expression name> ::= <identifier>
40. <identifier> ::= <alphabet> {{<digit>} {<alphabet>}{ **\_** }} | **\_** {{<digit>} {<alphabet>}{ **\_** }}

**Literal**

1. <literal> ::= <number literal> | <boolean literal> | <character literal> | <string literal> | <null literal>
2. <number literal> ::= **0** <zero number option> <integer type suffix> | <non zero digit> {<digit>} <non zero option>
3. <zero number option> ::= <hex numeral> {<hex numeral>} <integer type suffix> | <octal numeral> {<octal numeral>} <integer type suffix> | <floating-point literal> | 𝝴
4. <non-zero option> ::= <integer type suffix> | <floating-point literal>
5. <integer type suffix> ::= **l** | **L** | 𝝴
6. <digit> ::= **0** | <non zero digit>
7. <non zero digit> ::= **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9**
8. <hex numeral> ::= **x** <hex digit> | **X** <hex digit> | 𝝴
9. <hex digit> :: = **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **a** | **b** | **c** | **d** | **e** | **f** | **A** | **B** | **C** | **D** | **E** | **F**
10. <octal numeral> ::= **0** <octal digit>
11. <octal digit> ::= **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7**
12. <floating-point literal> ::= **.** {<digit>} <exponent part> <float type suffix> <digit> {<digit>} <exponent part> <float type suffix>
13. <exponent part> ::= <exponent indicator> <signed integer> | 𝝴
14. <exponent indicator> ::= **e** | **E**
15. <signed integer> ::= <sign> <digit> {<digit>}
16. <sign> ::= **+** | **-** | 𝝴
17. <float type suffix> ::= **f** | **F** | **d** | **D** | 𝝴
18. <boolean literal> ::= **true** | **false**
19. <character literal> ::= **'** <character literal option> **'**
20. <character literal option> ::= <single character> | <escape character>
21. <single character> ::= <input character> | **“**
22. <string literal> ::= **"** {<string character>} **"**
23. <alphabet> ::= **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** | **K** | **L** | **M** | **N** | **O** | **P** | **Q** | **R** | **S** | **T** | **U** | **V** | **W** | **X** | **Y** | **Z** | **a** | **b** | **d** | **e** | **f** | **g** | **h** | **i** | **j** | **k** | **l** | **m** | **n** | **o** | **p** | **q** | **r** | **s** | **t** | **u** | **v** | **w** | **x** | **y** | **z**
24. <string character> ::= <input character> | **‘** | <escape character>
25. <input character> ::= <alphabet> | <digit> | **!** | **#** | **$** | **%** | **&** | **(** | **)** | **\*** | **+** | **,** | **-** | **.** | **/** |**:** | **;** | **<** | **=** | **>** | **?** | **@** | **[** | **]** | **^** | **\_** | **`** | **{** | **|** | **}** | **~**
26. <all input character> ::= <input character> | **“** | **'** | **\**
27. <escape character> ::= **\t** | **\b** | **\n** | **\r** | **\f** | **\’** | **\”** | **\\**
28. <null literal> ::= **null**