I. SYNTACTIC ELEMENTS

- **A. Character Set**

<Characters> ::== <Letters> | <Digits> | <Special Characters>

<Letters> ::== <Uppercase_Letters> | <Lowercase_Letters>

<Uppercase_Letters> ::== 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

<Lowercase_Letters> ::== 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

<Decimal> ::== <Digits> | <Digits>{<Digits>}*.<Digits>{<Digits>}*

<Digits> ::== 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<Special_Characters> ::== . | + | - | * | / | \ | % | < | > | = | “ | ‘ | ` | : | ; | | | (|) | [|] | { | } | _ | ^ | ~ | & | $ | ! | ? | @ | #

<Whitespaces> ::== blank space | tab | carriage return | newline | form feed

- **B. Identifiers**

<Identifiers**>** ::== <Data_Types>? <Letters> <Letters>$^{+}$ | <Letter> {<Letter> | <Digit >}$^{+}$ | <Letter> (‘_’ | ‘-’)* <Digit>$^{+}$

- **C. Data Types**

<Data_Types> ::== int | float | double | char | string | bool

- **D. Operators Symbols**

<Operators**>** ::== <Assignment_Op> | <Arithmetic_Op> | <Unary_Op> | <Relational_Op> | <Logical_Op>

<Assignment_Op**>** ::== = | is | += | -= | *= | /= | %=

<Grouping_Op> ::== ( )

<Unary_Op> ::== ‘++’ | ‘--’ | ‘-’

<Arithmetic_Op> ::== ‘*’ | ‘/’ | ‘%’ | times | divideBy | mod | ‘+’ | ‘-’ | plus | minus

<Relational_Op> ::== ‘>’ | ‘<’ | ‘>=’ | ‘<=’ | ‘==’ | ‘!=’ | greaterThan | lessThan | greaterThanOrEqual | lessThanEqual | equalTo | notEqualTo

<Logical_Op> ::== ‘!’ | NOT | ‘&&’ | AND | ‘ ||’ | OR

<Ternary_Op> ::== <Identifiers> <Relational_Op> <Identifiers> ‘?’
<Identifiers> ‘:’ <Identifiers>

- **E. Keywords and Reserved words**

<Keywords> ::== AND | OR | NOT | plus | minus | times | divideBy | mod | equalTo | notEqualTo | greaterThan | greaterThanOrEqual | lessThan | lessThanOrEqual

<Res_words> ::== true | false | break | cont | for | do | while | if | else | switch | default | input | display | int | float | double | char | bool | define | void | return | is | var

- **F. Noise Words**

<Noise_Words> ::== ean | acter | inue | eger | ulus

- **G. Comments**

<Single_ln_cmmt> ::== ##<Characters>

<Query_cmmt> ::== #?<Characters>

<Multi_ln_cmmt> ::== #*<Characters>*#

<Multi_query_cmmt> ::== <???<Characters>???>

- **H. Delimiters and Brackets**

<Delimiters> ::== **;** | **,**

<Brackets> ::== (|) | { | }

- **Expressions**

<Expressions> ::= <Expressions> <Arithmetic_Operator> <Expression>

| <Expressions> <Assignment_Operator> <Expressions>

| <Identifiers> <Unary_Operator>

| <Unary_Operator> <Identifiers>

| <Identifiers>$^{+}$

| <Expressions> <Logical_Operator> <Expressions>

| <Expression> <Relational_Boolean> <Expressions>

| (<Expressions>)

| <Switch_Condition>

| <Declarations>

| <Identifiers>

| <Sign> <Constants>

| ! <Expression>

| <Identifiers> <Operators> (<const> | <Identifiers>)

II. SYNTAX

Production Rule:

NON-TERMINALS	**PRODUCTION RULES**
<program>	<program> ::== <declarations> <stmts>$^{+}$ | 'void';
<stmts>	<stmts> ::== (<Expressions>$^{+}$ | <Characters>$^{+}$)*
<declarations>	<declarations> ::== <declarations>$^{+}$
<declaration>	<declaration> ::== <ID_declaration>

<ID_declaration>	<Identifiers_declaration> ::== <Data_Types> <Identifiers>; | <Data_Types> <Identifier> = <Expressions>;
<param>	<param> ::== <Data_Types> <Identifier>
<stmt>	<stmt> ::== <Input_stmt> | <Output_stmt> | <Assignment_stmt> | <Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Input_stmt>	<Input_stmt> ::== <Identifiers> (“is” |” =”) input();
<Output_stmt>	<Output_stmt> ::== display(<char> | <string>); | display(<string> + <Identifiers>); | display(<string>) + <Identifiers> {+ <string> + <Identifiers>*}; | display(<Identifiers>);
<Assignment_stmt>	<Assignment_Op> ::==
<Conditional_Stmt>	<Conditional_Stmt> ::== **<**if_stmt**> | <**Switch_stmt**> |** <if_else_stmt> | <if_elseif_else_stms> | **<**nested_if_stmt**> | <**nested_if_stmt**> | <**nested_if_else_if_stmt**> |**
<if_stmt>	<if_stmt> ::== if (<Expressions>) {<stmts>$^{+}$;}
<if_else_stmt>	<if_else_stmt> ::== <if_stmt> + else {<stmts>$^{+}$}
<if_elseif_else_stmt>	<if_elseif_else_stmt> ::== <if_stmt> + else if (<Expressions>){<stmts>$^{+}$} else {<stmts>$^{+}$}
<Switch_stmt>	<Switch_stmt> ::== switch (<Expressions>) {<Case_stmt>}
<Case_stmt>	<Case_stmt> ::== <Case_part> <Case_stmt> | <Case_part>
<Case_part>	<Case_part> ::== case : <Expressions> {<stmts>$^{+}$} <break>; <Default_stmt>*;
<Default_stmt>	<Default_stmt> ::== default <stmts>$^{+}$;
<break>	<break> ::== break;
<Iterative_stmt>	<Iterative_stmt> ::== for ‘(’ (<assignment_stmt> | <declaration_stmt>) <ID> <Expressions> <int>; <Expressions> <ID> ‘)’ ‘{‘<stmt>’}’ | while ‘(‘<Expressions>’)’ {<stmt>}
<Return_stmt>	<return_stmt> ::== return <Digits>;
<expr>	<expr> ::== <Expressions> | <logical_or_expr> | <logical_and_expr> | <logical_not_expr> | <bool_expr>
<unary_expr>	<unary_expr> ::== ++<unary_expr> | --<unary_expr> | <unary_expr>++ | <unary_expr>-- | -<unary_exp>

<const>	<const> ::== <int> | <float> | <double> | <char> | <string> | <bool>
<int>	<integer> ::== [“-”] <Digits>$^{+}$
<float>	<float> ::== [“-”] <Digits>$^{+}$.<Digits>$^{+}$
<double>	<double> ::== [“-”] <Digits>$^{+}$ | [“-”] <Digits>$^{+}$.<Digits>$^{+}$
<char>	<char> ::== ‘<Uppercase_Letters>’ | ‘<Lowercase_Letters>’
<string>	<string> ::== “<<Uppercase_Letters>$^{+}$” | “<Lowercase_Letters>$^{+}$”
<bool>	<bool> ::== <true> | <false>

A. Input Statement

Grammar Rule:

<Input_stmts**>** ::== <Identifiers> <Assignmnet_Op> input();

Example:

num is input();

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

<program> => <smts>
<stmts> => <stmt>
<stmt> => <Input_stmts>
<Input_stmts> => <Identifiers> <Assignmner_Op> input();
=> num <Assignmner_Opt> input();

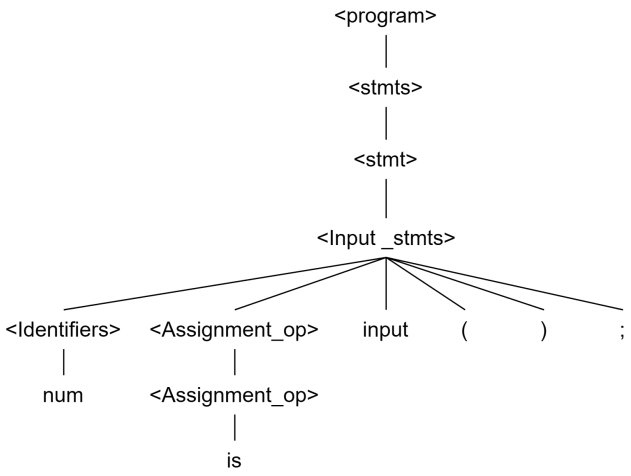
=> num is input();

Rightmost (Top-Down - Right-Left)

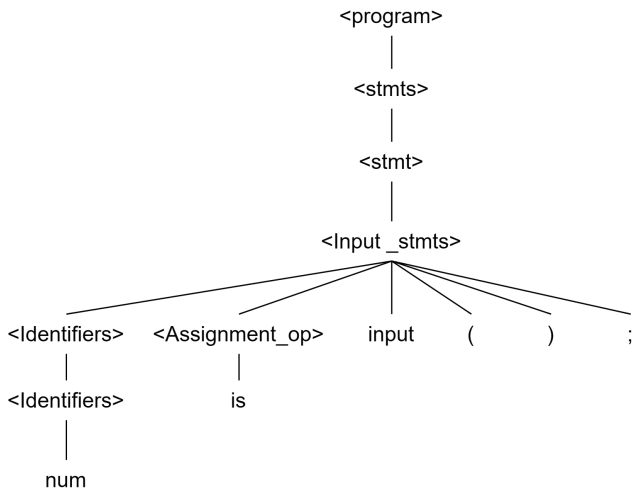
<program> => <smts>
<stmts> => <stmt>
<stmt> => <Input_stmts>
<Input_stmts> => <Identifiers> <Assignment_Op> input();
=> <Identifiers> is input();
=> num is input();

b. Parse tree

Leftmost (Top-Down - Left-Right)



Rightmost (Top-Down - Right-Left)



B. Output Statement

Grammar Rule:

<Output_Statement> ::== <display> (<Expressions> | <stmts>);

Example:

display (“Hello World!”);

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

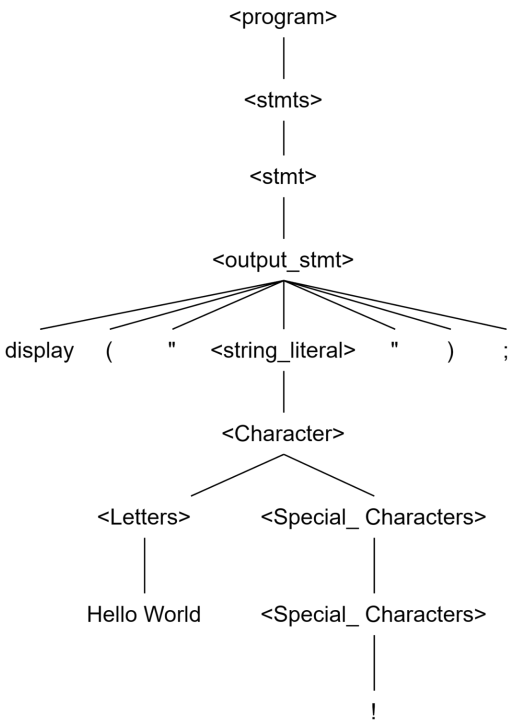
<program>	=> <smts>
<stmts>	=> <stmt>
<stmt>	=> <Output_stmts>
<Output_stmts>	=> <display> <stmts>;
	=> display <stmts>;
	=> display (“Hello World!”);

Rightmost (Top-Down - Right-Left)

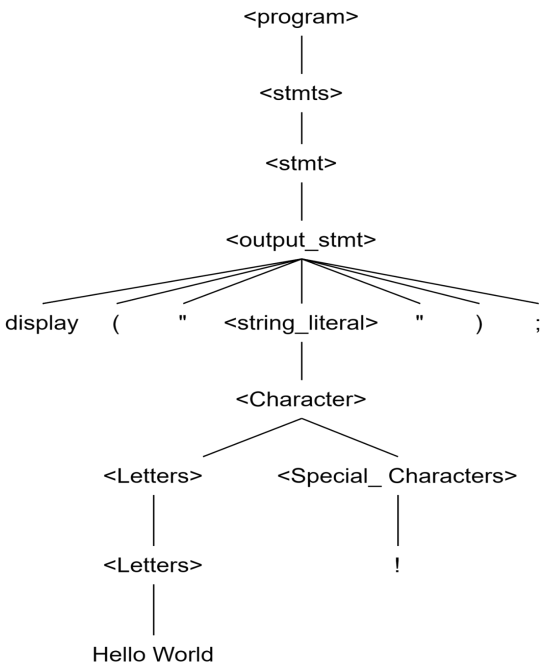
<program>	=> <smts>
<stmts>	=> <stmt>
<stmt>	=> <Output_stmts>
<Output_Statement>	=> <display> <stmts>;
	=> <display> (“Hello World!”);
	=> display (“Hello World!”);

b. Parse tree

Leftmost (Top-Down - Left-Right)



Rightmost (Top-Down - Right-Left)



C. Conditional Statements

<Conditional_Statements> **::==** **<If_Condition> | <Switch_Condition> |**
<If-Else_Condition> | <If-ElseIf-Else_Condition>

- **If Statements**

  Grammar Rules

  <If_Condition> **::==** if (<Boolean_Expression>) <Statements>;

  <If_Else_Condition> ::== if (<Boolean_Expression>) <Statements>;

  else <Statements>;

  <If_Elseif_Else_Condition> ::== if (<Boolean_Expression>) <Statements>;

  else if (<Boolean_Expression>) <Statements>;

  else <Statements>;

  <Nested_If_Condition> ::= if (<Boolean_Expression>)

  if (<Boolean_Expression>) <Statements>;

  <Nested_If_Else_Condition> ::= if (<Boolean_Expression>)

  if (<Boolean_Expression>) <Statements>;

  else <Statements>;

  else <Statements>;

  <Nested_If_Else_If_Condition> ::= if (<Boolean_Expression>)

  if (<Boolean_Expression>) <Statements>;

  else <Statements>;

  else if (<Boolean_Expression>)

  if (<Boolean_Expression>)

else if (<Boolean_Expression>) <Statements>;

else

if (<Boolean_Expression>) <Statements>;

else if (<Boolean_Expression>) <Statements>;

else <Statements>;

Example

- **if Condition**

```
if (grade greaterThanOrEqual 75) {
   display ("Congratulations you Passed");
}
```

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> |
<Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Conditional_Stmt> => **<**if_stmt**> | <**Switch_stmt**> |** <if_else_stmt> |
<if_elseif_else_stms> | **<**nested_if_stmt**> | <**nested_if_stmt**> |**
<nested_if_else_if_stmt**>**
<if_stmt**>** **=>** if (<Expressions>) {<stmts>$^{+}$;}
=> if (<Identifiers><Operators><const>) {<stmts>$^{+}$;}
=> if (<Identifiers><Operators><const>) {<stmts>}
=> if (grade <Operators> <const>){<stmts>}
=> if (grade <Relational_Op> <const>){<stmts>}
=> if (grade >= <const>) {<stmts>}
=> if (grade >= int) {<stmts>}
=> if (grade >= 75) {<stmts>}
=> if (grade >= 75) {display("Congratulations you Passed"); }

Rightmost (Top-Down - Right-Left)

<program> => <stmts>

<stmts> => <stmt>
<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> |
<Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Conditional_Stmt> => **<**if_stmt**> | <**Switch_stmt**> |** <if_else_stmt> |
<if_elseif_else_stms> | **<**nested_if_stmt**> | <**nested_if_stmt**> |**
<nested_if_else_if_stmt**>**
<if_stmt**>** **=>** if (<Expressions>) {<stmts>$^{+}$;}
=> if (<Expressions>) {<stmts>}
=> if (<Expressions>) {display("Congratulations you Passed"); }
=> if (<Identifiers> <Operators><const>)
{display("Congratulations you Passed"); }
=> if (<Identifiers> <Operators> int) {display("Congratulations
you Passed"); }
=> if (<Identifiers> <Operators> 75) {display("Congratulations
you Passed"); }
=> if (<Identifiers> <Relational_Op> 75)
{display("Congratulations you Passed"); }
=> if (<Identifiers> >= 75)
{display("Congratulations you Passed"); }
=> if (grade >= 75) {display("Congratulations you Passed"); }

b. Parse tree
Leftmost (Top-Down - Left-Right)
Rightmost (Top-Down - Right-Left)

- **if-else Condition**

```
int grade is 75;
if (grade greaterThanOrEqual 75) {
   display ("Congratulations you Passed" );
 }
else {
display ("Sorry, you failed" );
 }
```

a. Instantaneous Description
Leftmost (Top-Down - Left-Right)
<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> |
<Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Conditional_stmt> => <Conditional_Stmt> ::== **<**if_stmt**> | <**Switch_stmt**> |**
<if_else_stmt> | <if_elseif_else_stms> |

<nested_if_stmt> | <nested_if_stmt> |
<nested_if_else_if_stmt> | <if_elseif_else_stmt>

<if_else_stmt> => <if_stmt> + else{<stmts>}
=> if (<Expressions>) {<stmts>};
else{<stmts>}
=> if (<Identifiers><Operators><const>) {<stmts>}
else{<stmts>}
=> if (grade <Operators> <const>){<stmts>}
else{<stmts>}
=> if (grade <Relational_Op><const>){<stmts>;}
else{<stmts>}
=> if (grade >= <const>) {<stmts>} else{<stmts>}
=> if (grade >= int) {<Statements>} else{<stmts>}
=> if (grade >= 75) {<stmts>}
else{<stmts>}
=> if (grade >= 75) {display ("Congratulations you
Passed");} else{<stmts>}
=> if (grade >= 75) {display ("Congratulations you
Passed"); } else{display("Sorry, you failed");}

Rightmost (Top-Down - Right-Left)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> |
<Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Conditional_stmt> => <if_stmt> | <Switch_stmt> | <if_else_stmt> |
<if_elseif_else_stmt>
<if_else_stmt> => <if_stmt> + else{<stmts>}
=> if (<Expressions>) {<stmts>} else{<stmts>}
=> if (<Expressions>) {<stmts>}
else{display("Sorry you failed");}
=> if (<Expressions>) {display("Congratulations you
Passed!")} else{display("Sorry you failed");}
=> if (<Identifiers><Operators><const>)
{display("Congratulations you Passed!")}
else{display("Sorry you failed");}
=> if (<Identifiers><Operators><int>)
{display("Congratulations you Passed!")}

else{display(“Sorry you failed”);}
=> if (<Identifiers><Operators>75)
{display(“Congratulations you Passed!”)}
else{display(“Sorry you failed”);}
=> if (<Identifiers> <Relational_OP> 75) {
display(“Congratulations you Passed”)};
else{display(“Sorry, you failed”);}
=> if (<Identifiers> >= 75) { display(“Congratulations you Passed”)}; else{display(“Sorry, you failed”);}
=> if (<Identifiers> >= 75) { display(“Congratulations you Passed”)}; else{display(“Sorry, you failed”);}

b. Parse tree
Leftmost (Top-Down - Left-Right)
Rightmost (Top-Down - Right-Left)

- **if-elseif-else Condition**

```
int age is 21;
if (age greaterThanOrEqual 60) {
   display (“You are a Senior Citizen”);
}else if (age lessThanEqual 17) {
   display (“You are a Minor”);
} else {
   display (“You are an Adult”);
}
```

a. Instantaneous Description
Leftmost (Top-Down - Left-Right)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> | <Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Conditional_stmt> => <Conditional_Stmt> ::== **<**if_stmt**> | <**Switch_stmt**> |** <if_else_stmt> | <if_elseif_else_stms> | **<**nested_if_stmt**> | <**nested_if_stmt**> |** **<**nested_if_else_if_stmt**>** | <if_elseif_else_stmt>
<if_elseif_else_stmt> => <if_stmt> + else if (<Expressions>){<stmts>$^{+}$} else {<stmts>$^{+}$}
=> if(<Expressions>) {<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}
=> if(<Identifiers><Operators><const>) {<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}
=> if(age <Operators><const>) {<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}

=> if(age <Relational_Op><const>) {<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}
=> if(age >= <const>) {<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}
=> if(age >= <int>) {<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}
=> if(age >= 60) {<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}
=> if(age >= 60) {display("You are a Senior Citizen");}
else if(<Expressions>){<stmts>} else{<stmts>}
=> if(age >= 60) {display("You are a Senior Citizen");}
else if(<Identifiers><Operators><const>){<stmts>}
else{<stmts>}
=> if(age >= 60) {display("You are a Senior Citizen");}
else if(age <Operators><const>){<stmts>}
else{<stmts>}
=> if(age >= 60) {display("You are a Senior Citizen");}
else if(age <Relational_Op><const>){<stmts>}
else{<stmts>}
=> if(age >= 60) {display("You are a Senior Citizen");}
else if(age <= <const>){<stmts>}
else{<stmts>}
=> if(age >= 60) {display("You are a Senior Citizen");}
else if(age <= <int>){<stmts>}
else{<stmts>}
=> if(age >= 60) {display("You are a Senior Citizen");}
else if(age <= 17){<stmts>}else{<stmts>}
=> if(age >= 60) {display("You are a Senior Citizen");}
else if(age <= 17){display("You are a Minor");}
else{<stmts>}
=> if(age >= 60) {display("You are a Senior Citizen");}
else if(age <= 17){display("You are a Minor");}
else{display("You are an Adult");}

Rightmost (Top-Down - Right-Left)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> |
<Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Conditional_stmt> => <Conditional_Stmt> ::== **<**if_stmt**> | <**Switch_stmt**> |**
<if_else_stmt> | <if_elseif_else_stms> |

<nested_if_stmt> | <nested_if_stmt> |
<nested_if_else_if_stmt> | <if_elseif_else_stmt>

=> <if_stmt> + else if
(<Expressions>){<stmts>$^{+}$} else {<stmts>$^{+}$}

=> if(<Expressions>){<stmts>} else if
(<Expressions>){<stmts>}else{<stmts>}

=> if(<Expressions>){<stmts>} else if
(<Expressions>){<stmts>}else{display("You are an Adult");}

=> if(<Expressions>){<stmts>} else if
(<Expressions>){display("You are a Minor");}else{display("You are an Adult");}

=> if(<Expressions>){<stmts>} else if
(<Identifiers><Operators><const>){display("You are a Minor");}else{display("You are an Adult");}

=> if(<Expressions>){<stmts>} else if
(<Identifiers><Operators> int){display("You are a Minor");}else{display("You are an Adult");}

=> if(<Expressions>){<stmts>} else if
(<Identifiers><Operators> 17){display("You are a Minor");}else{display("You are an Adult");}

=> if(<Expressions>){<stmts>} else if
(<Identifiers><Relational_Op> 17){display("You are a Minor");}else{display("You are an Adult");}

=> if(<Expressions>){<stmts>} else if
(<Identifiers> <= 17){display("You are a Minor");}else{display("You are an Adult");}

=> if(<Expressions>){<stmts>} else if (age <= 17)
{display("You are a Minor");}else{display("You are an Adult");}

=> if(<Expressions>){display("You are a Senior Citizen");} else if (age <= 17){display("You are a Minor");}else{display("You are an Adult");}

=> if(<Identifiers><Operators><const>){display("You are a Senior Citizen");} else if (age <= 17){display("You are a Minor");}else{display("You are an Adult");}

=> if(<Identifiers><Operators><int>){display("You are a Senior Citizen");} else if (age <= 17){display("You are a Minor");}else{display("You are an Adult");}

=> if(<Identifiers><Operators> 60){display("You are a Senior Citizen");} else if (age <= 17){display("You are a Minor");}else{display("You are an Adult");}
=> if(<Identifiers><Relational_Op> 60){display("You are a Senior Citizen");} else if (age <= 17){display("You are a Minor");}else{display("You are an Adult");}
=> if(<Identifiers> >= 60){display("You are a Senior Citizen");} else if (age <= 17){display("You are a Minor");}else{display("You are an Adult");}
=> if(age >= 60){display("You are a Senior Citizen");} else if (age <= 17){display("You are a Minor");}else{display("You are an Adult");}

b. Parse tree
Leftmost (Top-Down - Left-Right)
Rightmost (Top-Down - Right-Left)

- **nested if**

```
if (a equalTo 5){
   if (b equalTo 10) {
      display ("The value of a is 5 and the value of b is 10.");
   }
}
```

a. Instantaneous Description
Leftmost (Top-Down - Left-Right)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> | <Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Conditional_Stmt> => **<if_stmt> | <Switch_stmt> |** <if_else_stmt> | <if_elseif_else_stms> | **<nested_if_stmt> | <nested_if_stmt> | <nested_if_else_if_stmt>**
<nested_if_stmt> => if (<Expressions>) {if (<Expressions>) {<stmts>$^{+}$;}}
=> if (<Identifiers><Operators><const>) {if (<Expressions>) {<stmts>$^{+}$;}}
=> if (a <Operators><const>) {if (<Expressions>) {<stmts>$^{+}$;}}
=> if (a <Relational_Op><const>) {if (<Expressions>) {<stmts>$^{+}$;}}

=> if (a equalTo <const>) {if
(<Expressions>) {<stmts>$^{+}$;}}
=> if (a equalTo int) {if
(<Expressions>) {<stmts>$^{+}$;}}
=> if (a equalTo 5) {if
(<Expressions>) {<stmts>$^{+}$;}}
=> if (a equalTo 5) {if
(<Identifiers><Operators><const>) {<stmts>$^{+}$;}}
=> if (a equalTo 5) {if
(b <Operators><const>) {<stmts>$^{+}$;}}
=> if (a equalTo 5) {if
(b <Relational_Op> <const>) {<stmts>$^{+}$;}}
=> if (a equalTo 5) {if
(b equalTo <const>) {<stmts>$^{+}$;}}
=> if (a equalTo 5) {if
(b equalTo int) {<stmts>$^{+}$;}}
=> if (a equalTo 5) {if
(b equalTo 10) {<stmts>$^{+}$;}}
=> if (a equalTo 5) {if
(b equalTo 10) {<stmts>}}
=> if (a equalTo 5) {if
(b equalTo 10) {("The value of a is 5 and the value of b is 10.");}}

Rightmost (Top-Down - Right-Left)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> | <Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Conditional_Stmt> => **<**if_stmt**> | <**Switch_stmt**> |** <if_else_stmt> | <if_elseif_else_stms> | **<**nested_if_stmt**> | <**nested_if_stmt**> | <**nested_if_else_if_stmt**>**
<nested_if_stmt**>** => if (<Expressions>) {if (<Expressions>) {<stmts>$^{+}$;}}
=> if (<Expressions>) {if (<Expressions>) {<stmts>}}
=> if (<Expressions>) {if (<Expressions>) {("The value of a is 5 and the value of b is 10.");}}
=> if (<Expressions>) {if
(<Identifiers><Operators><const>) {("The value of a is 5 and the value of b is 10.");}}
=> if (<Expressions>) {if

(<Identifiers><Operators>int) {(“The value of
a is 5 and the value of b is 10.”);}}
=> if (<Expressions>) {if
(<Identifiers><Operators> 10) {(“The value of
a is 5 and the value of b is 10.”);}}
=> if (<Expressions>) {if
(<Identifiers><Relational_Op> 10) {(“The value of
a is 5 and the value of b is 10.”);}}
=> if (<Expressions>) {if
(<Identifiers> equalTo 10) {(“The value of
a is 5 and the value of b is 10.”);}}
=> if (<Expressions>) {if
(b equalTo 10) {(“The value of
a is 5 and the value of b is 10.”);}}
=> if (<Identifiers><Operators><const>) {if
(b equalTo 10) {(“The value of
a is 5 and the value of b is 10.”);}}
=> if (<Identifiers><Operators> int) {if
(b equalTo 10) {(“The value of
a is 5 and the value of b is 10.”);}}
=> if (<Identifiers><Operators> 5) {if
(b equalTo 10) {(“The value of
a is 5 and the value of b is 10.”);}}
=> if (<Identifiers><Relational_Op> 5) {if
(b equalTo 10) {(“The value of
a is 5 and the value of b is 10.”);}}
=> if (<Identifiers> equalTo 5) {if
(b equalTo 10) {(“The value of
a is 5 and the value of b is 10.”);}}
=> if (a equalTo 5) {if
(b equalTo 10) {(“The value of
a is 5 and the value of b is 10.”);}}

b. Parse tree
Leftmost (Top-Down - Left-Right)
Rightmost (Top-Down - Right-Left)

- **nested if else**

```
int value is 85;
if (value greaterThanOrEqual 75) {
   if (value equalTo 85) {
```

```
        display ("The value is exactly 85");
    } else {
            display ("The value is more than 85");
            }
    } else {
            display ("The value is less than 85");
}
```

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

<program> => <stmts>

<stmts> => <stmt>

<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> | <Conditional_stmt> | <Iterative_stmt> | <Return_stmt>

<Conditional_Stmt> => **<**if_stmt**> | <**Switch_stmt**> |** <if_else_stmt> | <if_elseif_else_stms> | **<**nested_if_stmt**> | <**nested_if_stmt**> | <**nested_if_else_if_stmt**>**

<nested_if_stmt**>** **=>** if (<Expressions>) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>$^{+}$;}} else {<stmts>$^{+}$;}}

=> if (<Identifiers><Operators><const>) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>$^{+}$;}} else {<stmts>$^{+}$;}}

=> if (value <Operators><const>) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>$^{+}$;}} else {<stmts>$^{+}$;}}

=> if (value <Relational_Op> <const>) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>$^{+}$;}} else {<stmts>$^{+}$;}}

=> if (value greaterThanOrEqual <const>) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>$^{+}$;}} else {<stmts>$^{+}$;}}

=> if (value greaterThanOrEqual int) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>$^{+}$;}} else {<stmts>$^{+}$;}}

=> if (value greaterThanOrEqual 75) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>$^{+}$;}} else {<stmts>$^{+}$;}}

=> if (value greaterThanOrEqual 75) {if (<Identifiers><Operators><const>) {<stmts>$^{+}$;}else

{<stmts>+;}} else {<stmts>+;}}

=> if (value greaterThanOrEqual 75) {if
(value <Operators><const>) {<stmts>+;}else
{<stmts>+;}} else {<stmts>+;}}

=> if (value greaterThanOrEqual 75) {if
(value <Relational_Op> <const>) {<stmts>+;}else
{<stmts>+;}} else {<stmts>+;}}

=> if (value greaterThanOrEqual 75) {if
(value equalTo <const>) {<stmts>+;}else
{<stmts>+;}} else {<stmts>+;}}

=> if (value greaterThanOrEqual 75) {if
(value equalTo int) {<stmts>+;}else
{<stmts>+;}} else {<stmts>+;}}

=> if (value greaterThanOrEqual 75) {if
(value equalTo 85) {<stmts>+;}else
{<stmts>+;}} else {<stmts>+;}}

=> if (value greaterThanOrEqual 75) {if
(value equalTo 85) {<stmts>}else
{<stmts>+;}} else {<stmts>+;}}

=> if (value greaterThanOrEqual 75) {if
(value equalTo 85) {display (“The value is exactly
85”);}else {<stmts>+;}} else {<stmts>+;}}

=> if (value greaterThanOrEqual 75) {if
(value equalTo 85) {display (“The value is exactly
85”);}else {<stmts>}} else {<stmts>+;}}

=> if (value greaterThanOrEqual 75) {if
(value equalTo 85) {display (“The value is exactly
85”);}else {display (“The value is more than 85”);}}
else {<stmts>+;}}

=> if (value greaterThanOrEqual 75) {if
(value equalTo 85) {display (“The value is exactly
85”);}else {display (“The value is more than 85”);}}
else {<stmts>}}

=> if (value greaterThanOrEqual 75) {if
(value equalTo 85) {display (“The value is exactly
85”);}else {display (“The value is more than 85”);}}
else { display (“The value is less than 85”);}}

Rightmost (Top-Down - Right-Left)

<program> => <stmts>

<stmts> => <stmt>

<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> | <Conditional_stmt> | <Iterative_stmt> | <Return_stmt>

<Conditional_Stmt> => **<**if_stmt**> | <**Switch_stmt**> |** <if_else_stmt> | <if_elseif_else_stms> | **<**nested_if_stmt**> | <**nested_if_stmt**> | <**nested_if_else_if_stmt**>**

<nested_if_stmt**>** **=>** if (<Expressions>) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>$^{+}$;}} else {<stmts>$^{+}$;}}

=> if (<Expressions>) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>$^{+}$;}} else {<stmts>}}

=> if (<Expressions>) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>$^{+}$;}} else {display (“The value is less than 85”);}}

=> if (<Expressions>) {if (<Expressions>) {<stmts>$^{+}$;}else {<stmts>}} else {display (“The value is less than 85”);}}

=> if (<Expressions>) {if (<Expressions>) {<stmts>$^{+}$;}else {display (“The value is more than 85”);}} else {display (“The value is less than 85”);}}

=> if (<Expressions>) {if (<Expressions>) {<stmts>}else {display (“The value is more than 85”);}} else {display (“The value is less than 85”);}}

=> if (<Expressions>) {if (<Expressions>) {display (“The value is exactly 85”);}else {display (“The value is more than 85”);}} else {display (“The value is less than 85”);}}

=> if (<Expressions>) {if (<Identifiers><Operators><const>) {display (“The value is exactly 85”);}else {display (“The value is more than 85”);}} else {display (“The value is less than 85”);}}

=> if (<Expressions>) {if (<Identifiers><Operators> int) {display (“The value is exactly 85”);}else {display (“The value is more than 85”);}} else {display (“The value is less than 85”);}}

=> if (<Expressions>) {if (<Identifiers><Operators> 85) {display (“The value is exactly 85”);}else {display (“The value is more than 85”);}} else {display (“The

value is less than 85”);}}

=> if (<Expressions>) {if
(<Identifiers> <Relational_Op> 85)
{display (“The value is exactly 85”);}else {display
(“The value is more than 85”);}} else {display (“The
value is less than 85”);}}

=> if (<Expressions>) {if (<Identifiers> equalTo 85)
{display (“The value is exactly 85”);}else {display
(“The value is more than 85”);}} else {display (“The
value is less than 85”);}}

=> if (<Expressions>) {if (value equalTo 85)
{display (“The value is exactly 85”);}else {display
(“The value is more than 85”);}} else {display (“The
value is less than 85”);}}

=> if (<Identifiers><Operators><const>) {if (value
equalTo 85) {display (“The value is exactly 85”);}else
{display (“The value is more than 85”);}} else {display
(“The value is less than 85”);}}

=> if (<Identifiers><Operators> int) {if (value
equalTo 85) {display (“The value is exactly 85”);}else
{display (“The value is more than 85”);}} else {display
(“The value is less than 85”);}}

=> if (<Identifiers><Operators> 75) {if (value
equalTo 85) {display (“The value is exactly 85”);}else
{display (“The value is more than 85”);}} else {display
(“The value is less than 85”);}}

=> if (<Identifiers><Relational_Op> 75) {if (value
equalTo 85) {display (“The value is exactly 85”);}else
{display (“The value is more than 85”);}} else {display
(“The value is less than 85”);}}

=> if (<Identifiers> greaterThanOrEqual 75) {if (value
equalTo 85) {display (“The value is exactly 85”);}else
{display (“The value is more than 85”);}} else {display
(“The value is less than 85”);}}

=> if (value greaterThanOrEqual 75) {if (value
equalTo 85) {display (“The value is exactly 85”);}else
{display (“The value is more than 85”);}} else {display
(“The value is less than 85”);}}

b. Parse tree

Leftmost (Top-Down - Left-Right)
Rightmost (Top-Down - Right-Left)

- **nested if else if**

```
float bmi is 21.5
if (bmi lessThanEqual 18.4) {
   if (bmi equalTo 18.4) {
      display ("You are underweight");
   } else {
     display ("You should eat more");
         }
}

else if (bmi lessThanEqual 24.9) {
          if (bmi equalTo 18.5) {
             display ("You are in normal weight");
          } else if (bmi lessThan 25.0) {
                    display ("Keep up the healthy weight");
                  }
          }
          else {
            if (bmi greaterThanOrEqual 25.0) {
                     display ("You are overweight");
          }else if (bmi lessThan 30.0) {
                    display ("Still overweight");
          } else {
                    display ("You are obese");
                   }
           }
```

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> |
<Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Conditional_stmt> => <Conditional_Stmt> ::== **<**if_stmt**> | <**Switch_stmt**> |**
<if_else_stmt> | <if_elseif_else_stms> |
<nested_if_stmt**> | <**nested_if_stmt**> |**
<nested_if_else_if_stmt**> |** <if_elseif_else_stmt>

<nested_if_else_if_stmt> => if(<Expressions>){
if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(<Identifiers><Operators><const>){
if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(bmi <Operators><const>){
if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(bmi <Relational_Op><const>){
if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(bmi <= <const>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(bmi <= <float>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(bmi <= 18.4)
{if(<Identifiers><Operators><const>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(bmi <= 18.4){if(bmi <Operators><const>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(bmi <= 18.4){if(bmi <Relational_Op><const>)
{<stmts>}else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(bmi <= 18.4){if(bmi equalTo <const>)
{<stmts>}else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(bmi <= 18.4){if(bmi equalTo <float>)
{<stmts>}else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{<stmts>}else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if

(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(<stmts>)}}
else if(<Expressions>){if(<Expressions>)
{<stmts>}else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(display("You
should eat more");)}}else if(<Expressions>)
{if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(display("You
should eat more");)}}
else if(<Identifiers><Operators><const>)
{if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(display("You
should eat more");)}}
else if(bmi <Operators><const>)
{if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(display("You
should eat more");)}}
else if(bmi <Relational_Op><const>)
{if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}
else if(bmi <= <const>){if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}
else if(bmi <= <float>){if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}
else if(bmi <= 24.9){if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(<Identifiers><Operators><const>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi <Operators><const>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi <Relational_Op><const>){<stmts>}
else if(<Expressions>){<stmts>}}else{

if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(display("You should eat more");)}}else if(bmi <= 24.9)
{if(bmi equalTo <const>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(display("You should eat more");)}}else if(bmi <= 24.9)
{if(bmi equalTo <float>){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(display("You should eat more");)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){<stmts>}
else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(display("You should eat more");)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display("You are in Normal Weight");}else if(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(display("You should eat more");)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display("You are in Normal Weight");}else
if(<Identifiers><Operators><const>){<stmts>}}
else{if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display("You are Underweight");}else{(display("You

should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal
Weight”);}else if(bmi <Operators><const>){<stmts>}}
else{if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You
should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal
Weight”);}else if(bmi <Relational_Op>
<const>){<stmts>}}else{if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You
should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal
Weight”);}else if(bmi < <const>){<stmts>}}
else{if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You
should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal
Weight”);}else if(bmi < <float>){<stmts>}}
else{if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You
should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal
Weight”);}else if(bmi < 25.0){<stmts>}}
else{if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You
should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal
Weight”);}else if(bmi < 25.0){display(“Keep up the
Healthy Weight”);}}else{if(<Expressions>){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(<Identifiers><Operators><const>){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi <Operators><const>){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi <Relational_Op><const>){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= <const>){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= <float>){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= 25.0){<stmts>}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= 25.0){display(“You are Overweight”);}
else if(<Expressions>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= 25.0){display(“You are Overweight”);}
else if(<Identifiers><Operators><const>){<stmts>}
else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= 25.0){display(“You are Overweight”);}
else if(bmi <Operators><const>){<stmts>}
else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}

else{if(bmi >= 25.0){display(“You are Overweight”);}
else if(bmi <Relational_Op><const>){<stmts>}
else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= 25.0){display(“You are Overweight”);}
else if(bmi < <const>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= 25.0){display(“You are Overweight”);}
else if(bmi < <float>){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= 25.0){display(“You are Overweight”);}
else if(bmi < 30.0){<stmts>} else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= 25.0){display(“You are Overweight”);}
else if(bmi < 30.0){display(“Still Overweight”);}
else{<stmts>}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4)
{display(“You are Underweight”);}else{(display(“You should eat more”);)}}else if(bmi <= 24.9)
{if(bmi equalTo 18.5){display(“You are in Normal

Weight”);}else if(bmi < 25.0){display(“Keep up the Healthy Weight”);}}
else{if(bmi >= 25.0){display(“You are Overweight”);}
else if(bmi < 30.0){display(“Still Overweight”);}
else{display(“You are Obese”);}}

Rightmost (Top-Down - Right-Left)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Input_stmt> | <Output_stmt> | <Assignment_stmt> | <Conditional_stmt> | <Iterative_stmt> | <Return_stmt>
<Conditional_Stmt> => **<**if_stmt**> | <**Switch_stmt**> |** <if_else_stmt> | <if_elseif_else_stmt>
<nested_if_else_if_stmt**>** **=>** if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{<stmts>}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){<stmts>} else{display(“You are Obese”);}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Expressions>){display(“Still Overweight”);}
else{display(“You are Obese”);}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Identifiers><Operators><const>)
{display(“Still Overweight”);}

else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Identifiers><Operators><float>)
{display(“Still Overweight”);}
else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Identifiers><Operators> 30.0)
{display(“Still Overweight”);}
else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Identifiers><Relational_Op> 30.0)
{display(“Still Overweight”);}
else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(<Identifiers> < 30.0)
{display(“Still Overweight”);}
else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){<stmts>} else
if(bmi < 30.0 {display(“Still Overweight”);}
else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Expressions>){display("You are Overweight");}
else if(bmi < 30.0 {display("Still Overweight");}
else{display("You are Obese");}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Identifiers><Operators><const>){display("You are
Overweight");}else if(bmi < 30.0 {display("Still
Overweight");}else{display("You are Obese");}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Identifiers><Operators><float>){display("You are
Overweight");}else if(bmi < 30.0 {display("Still
Overweight");}else{display("You are Obese");}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Identifiers><Operators> 25.0){display("You are
Overweight");}else if(bmi < 30.0 {display("Still
Overweight");}else{display("You are Obese");}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Identifiers><Relational_Op> 25.0){display("You
are Overweight");}else if(bmi < 30.0 {display("Still
Overweight");}else{display("You are Obese");}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Expressions>){<stmts>}}else{
if(<Identifiers> >= 25.0){display("You

are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>} else{(<stmts>)}} else if(<Expressions>){ if(<Expressions>){<stmts>}else if (<Expressions>){<stmts>}}else{ if(<Identifiers> >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>} else{(<stmts>)}} else if(<Expressions>){ if(<Expressions>){<stmts>}else if (<Expressions>){<stmts>}}else{ if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>} else{(<stmts>)}} else if(<Expressions>){ if(<Expressions>){<stmts>}else if (<Expressions>){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>} else{(<stmts>)}} else if(<Expressions>){ if(<Expressions>){<stmts>}else if (<Identifiers><Operators><const>){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>} else{(<stmts>)}} else if(<Expressions>){ if(<Expressions>){<stmts>}else if (<Identifiers><Operators><float>){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Identifiers><Operators> 25.0){display(“Keep up
the healthy weight”);}}else{if(bmi >=
25.0){display(“You are Overweight”);}else if(bmi <
30.0 {display(“Still Overweight”);}else{display(“You
are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Identifiers><Relational_Op> 25.0){display(“Keep up
the healthy weight”);}}else{if(bmi >=
25.0){display(“You are Overweight”);}else if(bmi <
30.0 {display(“Still Overweight”);}else{display(“You
are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(<Identifiers> < 25.0){display(“Keep up the healthy
weight”);}}else{if(bmi >= 25.0){display(“You are
Overweight”);}else if(bmi < 30.0 {display(“Still
Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){<stmts>}else if
(bmi < 25.0){display(“Keep up the healthy
weight”);}}else{if(bmi >= 25.0){display(“You are
Overweight”);}else if(bmi < 30.0 {display(“Still
Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Expressions>){display(“You are in normal
weight”);}else if (bmi < 25.0){display(“Keep up the
healthy weight”);}}else{if(bmi >= 25.0){display(“You
are Overweight”);}else if(bmi < 30.0 {display(“Still
Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Identifiers><Operators><const>){display("You are
in normal weight");}else if (bmi < 25.0)
{display("Keep up the healthy weight");}}
else{if(bmi >= 25.0){display("You are
Overweight");}else if(bmi < 30.0 {display("Still
Overweight");}else{display("You are Obese");}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Identifiers><Operators><float>){display("You are
in normal weight");}else if (bmi < 25.0)
{display("Keep up the healthy weight");}}
else{if(bmi >= 25.0){display("You are
Overweight");}else if(bmi < 30.0 {display("Still
Overweight");}else{display("You are Obese");}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Identifiers><Operators> 18.5){display("You are
in normal weight");}else if (bmi < 25.0)
{display("Keep up the healthy weight");}}
else{if(bmi >= 25.0){display("You are
Overweight");}else if(bmi < 30.0 {display("Still
Overweight");}else{display("You are Obese");}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Identifiers><Relational_Op> 18.5){display("You
are in normal weight");}else if (bmi < 25.0)
{display("Keep up the healthy weight");}}
else{if(bmi >= 25.0){display("You are
Overweight");}else if(bmi < 30.0 {display("Still
Overweight");}else{display("You are Obese");}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Expressions>){
if(<Identifiers> equalTo 18.5){display("You
are in normal weight");}else if (bmi < 25.0)
{display("Keep up the healthy weight");}}
else{if(bmi >= 25.0){display("You are
Overweight");}else if(bmi < 30.0 {display("Still
Overweight");}else{display("You are Obese");}}

```
=> if(<Expressions>){if(<Expressions>){<stmts>}
    else{(<stmts>)}} else if(<Expressions>){
    if(bmi equalTo 18.5){display("You
    are in normal weight");}else  if (bmi < 25.0)
    {display("Keep up the  healthy weight");}}
    else{if(bmi >= 25.0){display("You are
    Overweight");}else if(bmi < 30.0 {display("Still
    Overweight");}else{display("You are Obese");}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
    else{(<stmts>)}} else
    if(<Identifiers><Operators><const>){
    if(bmi equalTo 18.5){display("You
    are in normal weight");}else  if (bmi < 25.0)
    {display("Keep up the  healthy weight");}}
    else{if(bmi >= 25.0){display("You are
    Overweight");}else if(bmi < 30.0 {display("Still
    Overweight");}else{display("You are Obese");}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
    else{(<stmts>)}} else
    if(<Identifiers><Operators><float>){
    if(bmi equalTo 18.5){display("You
    are in normal weight");}else  if (bmi < 25.0)
    {display("Keep up the  healthy weight");}}
    else{if(bmi >= 25.0){display("You are
    Overweight");}else if(bmi < 30.0 {display("Still
    Overweight");}else{display("You are Obese");}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
    else{(<stmts>)}} else
    if(<Identifiers><Operators> 24.9){
    if(bmi equalTo 18.5){display("You
    are in normal weight");}else  if (bmi < 25.0)
    {display("Keep up the  healthy weight");}}
    else{if(bmi >= 25.0){display("You are
    Overweight");}else if(bmi < 30.0 {display("Still
    Overweight");}else{display("You are Obese");}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
    else{(<stmts>)}} else
    if(<Identifiers><Relational_Op> 24.9){
    if(bmi equalTo 18.5){display("You
    are in normal weight");}else  if (bmi < 25.0)
```

{display(“Keep up the healthy weight”);}}
else{if(bmi >= 25.0){display(“You are
Overweight”);}else if(bmi < 30.0 {display(“Still
Overweight”);}else{display(“You are Obese”);}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(<Identifiers> <= 24.9){
if(bmi equalTo 18.5){display(“You
are in normal weight”);}else if (bmi < 25.0)
{display(“Keep up the healthy weight”);}}
else{if(bmi >= 25.0){display(“You are
Overweight”);}else if(bmi < 30.0 {display(“Still
Overweight”);}else{display(“You are Obese”);}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(<stmts>)}} else if(bmi <= 24.9){
if(bmi equalTo 18.5){display(“You
are in normal weight”);}else if (bmi < 25.0)
{display(“Keep up the healthy weight”);}}
else{if(bmi >= 25.0){display(“You are
Overweight”);}else if(bmi < 30.0 {display(“Still
Overweight”);}else{display(“You are Obese”);}}
=> if(<Expressions>){if(<Expressions>){<stmts>}
else{(display(“You should eat more”);)}} else if(bmi
<= 24.9){ if(bmi equalTo 18.5){display(“You
are in normal weight”);}else if (bmi < 25.0)
{display(“Keep up the healthy weight”);}}
else{if(bmi >= 25.0){display(“You are
Overweight”);}else if(bmi < 30.0 {display(“Still
Overweight”);}else{display(“You are Obese”);}}
=> if(<Expressions>){if(<Expressions>){display(“You are
underweight”);} else{(display(“You should eat
more”);)}} else if(bmi <= 24.9){ if(bmi equalTo 18.5)
{display(“You are in normal weight”);}else if (bmi <
25.0){display(“Keep up the healthy weight”);}}
else{if(bmi >= 25.0){display(“You are
Overweight”);}else if(bmi < 30.0 {display(“Still
Overweight”);}else{display(“You are Obese”);}}
=> if(<Expressions>){if(<Identifiers><Operators><const>)
{display(“You are underweight”);} else{(display(“You
should eat more”);)}} else if(bmi <= 24.9){if(bmi
equalTo 18.5) {display(“You are in normal

weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Identifiers><Operators><float>) {display(“You are underweight”);} else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Identifiers><Operators> 18.4) {display(“You are underweight”);} else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Identifiers><Relational_Op> 18.4) {display(“You are underweight”);} else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(<Identifiers> equalTo 18.4) {display(“You are underweight”);} else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Expressions>){if(bmi equalTo 18.4) {display(“You are underweight”);}

else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Identifiers><Operators><const>){if(bmi equalTo 18.4){display(“You are underweight”);} else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Identifiers><Operators><float>){if(bmi equalTo 18.4){display(“You are underweight”);} else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Identifiers><Operators> 18.4){if(bmi equalTo 18.4){display(“You are underweight”);} else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Identifiers><Relational_Op> 18.4){if(bmi equalTo 18.4){display(“You are underweight”);} else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >=

25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(<Identifiers> <= 18.4){if(bmi equalTo 18.4) {display(“You are underweight”);} else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

=> if(bmi <= 18.4){if(bmi equalTo 18.4) {display(“You are underweight”);} else{(display(“You should eat more”);)}} else if(bmi <= 24.9){if(bmi equalTo 18.5) {display(“You are in normal weight”);}else if (bmi < 25.0){display(“Keep up the healthy weight”);}}else{if(bmi >= 25.0){display(“You are Overweight”);}else if(bmi < 30.0 {display(“Still Overweight”);}else{display(“You are Obese”);}}

b. Parse tree
Leftmost (Top-Down - Left-Right)
Rightmost (Top-Down - Right-Left)

- **Switch Condition Statement**
  Grammar Rules
  <Switch_Condition> ::== ::== switch (<expr>) {<Case_stmt>; <break>;}

  Example:

```
int yearLevel;
switch (yearLevel){
case 1: display (“You are freshman”);
        break;
case 2: display (“You are sophomore”);
        break;
case 3: display (“You are junior”);
```

```
            break;
        case 4: display (“You are senior”);
            break;
        default: display (“Invalid input”);
            break;
        }
```

a. Instantaneous Description
   Leftmost (Top-Down - Left-Right)
   Rightmost (Top-Down - Right-Left)
b. Parse tree
   Leftmost (Top-Down - Left-Right)
   Rightmost (Top-Down - Right-Left)

D. Iterative Statement

<Iterative_stmt> ::== <for_stmt> | <nested_for_stmt> | <while_stmt> | <nested_while_stmt>

- **For loop**

Grammar Rules

<for_stmt> => for <Expressions>* {<stmts>}

Example

```
for (int it is 1; it lessThan 5; ++it){
    display (“Hello!”);
}
```

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <for_stmt>
<for_stmt> => for (<Expressions>*) {<stmts>$^{+}$}
=> for (<Expressions>; <Expressions>; <Expressions>) {<stmts>}
=> for (<Identifiers> <Assignment_Op> <const>; <Expressions>; <Expressions>) {<stmts>}
=> for (<Data_Types> <Letters> <Assignment_Op> <const>; <Expressions>; <Expressions>) {<stmts>}
=> for (int <Letters> <Assignment_Op> <const>; <Expressions>; <Expressions>) {<stmts>}
=> for (int it <Assignment_Op> <const>; <Expressions>; <Expressions>) {<stmts>}
=> for (int it is <const>; <Expressions>; <Expressions>) {<stmts>}

=> for (int it is <Digits>; <Expressions>; <Expressions>) {<stmts>}
=> for (int it is 1; <Expressions>; <Expressions>) {<stmts>}
=> for (int it is 1; <Identifiers> <Relational_Op> <const>; <expr>)
{<stmts>}
=> for (int it is 1; it <Relational_Op> <const>; <expr>) {<stmts>}
=> for (int it is 1; it lessThan <const>; <expr>) {<stmts>}
=> for (int it is 1; it lessThan <Digits>; <expr>) {<stmts>}
=> for (int it is 1; it lessThan 5; <expr>) {<stmts>}
=> for (int it is 1; it lessThan 5; <Unary_Op> <Identifiers>) {<stmts>}
=> for (int it is 1; it lessThan 5; ++ <Identifiers>) {<stmts>}
=> for (int it is 1; it lessThan 5; ++it) {<stmts>}
=> for (int it is 1; it lessThan 5; ++it) {display <stmts>;}
=> for (int it is 1; it lessThan 5; ++it) {display Hello!;}

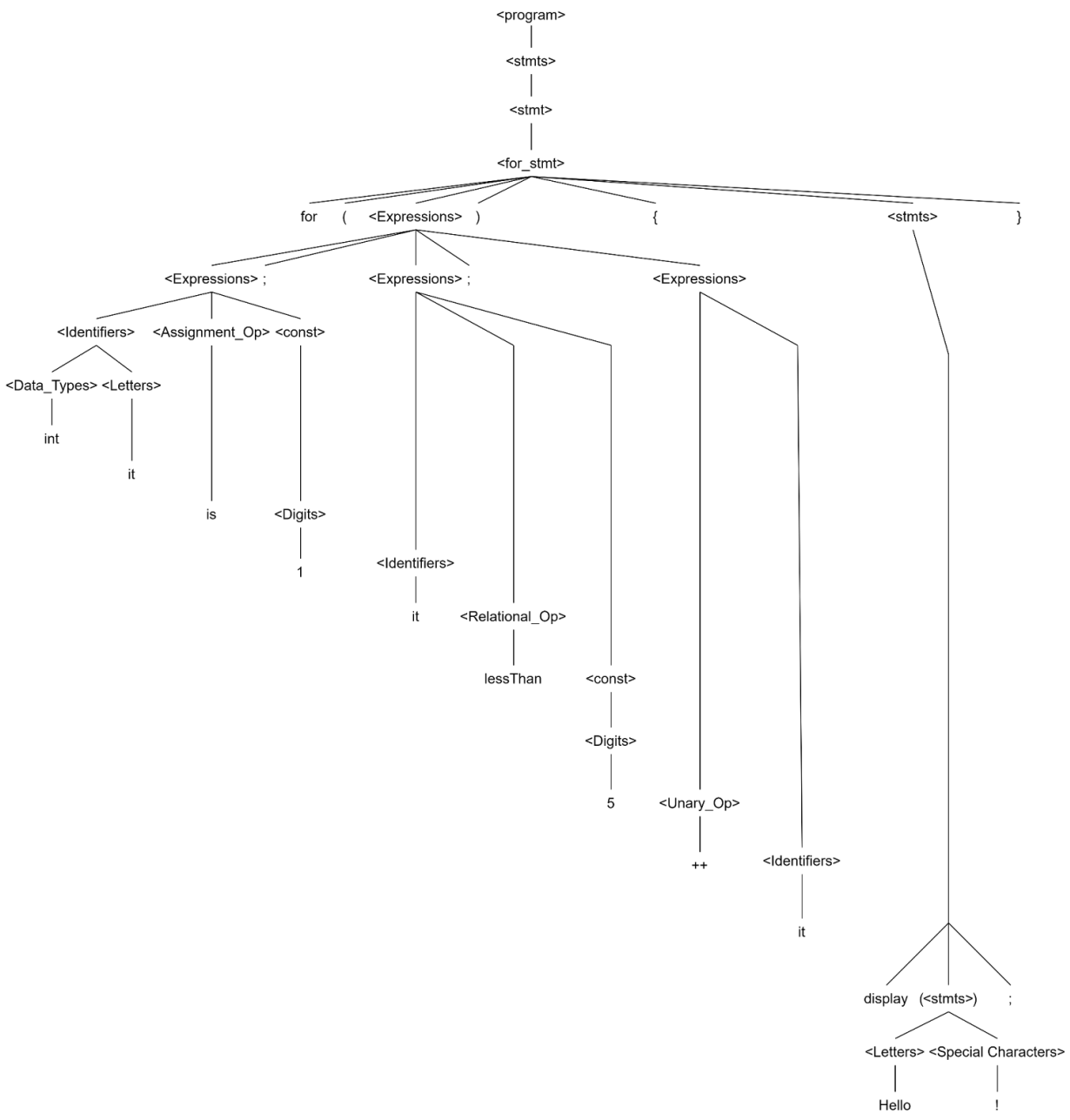
Rightmost (Top-Down - Right-Left)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <for_stmt>
<for_stmt> => for (<Expressions>*) {<stmts>$^{+}$}
=> for (<Expressions>*) {display <stmts>;}
=> for (<Expressions>*) {display (“Hello!”);}
=> for (<Expressions>; <Expressions>; <Expressions>) {display
(“Hello!”);}
=> for (<Expressions>; <Expressions>; <Unary_Op><Identifiers>)
{display (“Hello!”);}
=> for (<Expressions>; <Expressions>; <Unary_Op> it) {display
(“Hello!”);}
=> for (<Expressions>; <Identifiers> <Relational_Op> <consts>; ++it)
{display (“Hello!”);}
=> for (<Expressions>; <Identifiers> <Relational_Op> <Digits>; ++it)
{display (“Hello!”);}
=> for (<Expressions>; <Identifiers> <Relational_Op> 5; ++it) {display
(“Hello!”);}
=> for (<Expressions>; <Identifiers> lessThan 5; ++it) {display
(“Hello!”);}
=> for (<Expressions>; it lessThan 5; ++it) {display (“Hello!”);}
=> for (<Identifiers> <Assignment_Op> <const>; it lessThan 5; ++it)
{display (“Hello!”);}
=> for (<Identifiers> <Assignment_Op> <Digits>; it lessThan 5; ++it)
{display (“Hello!”);}

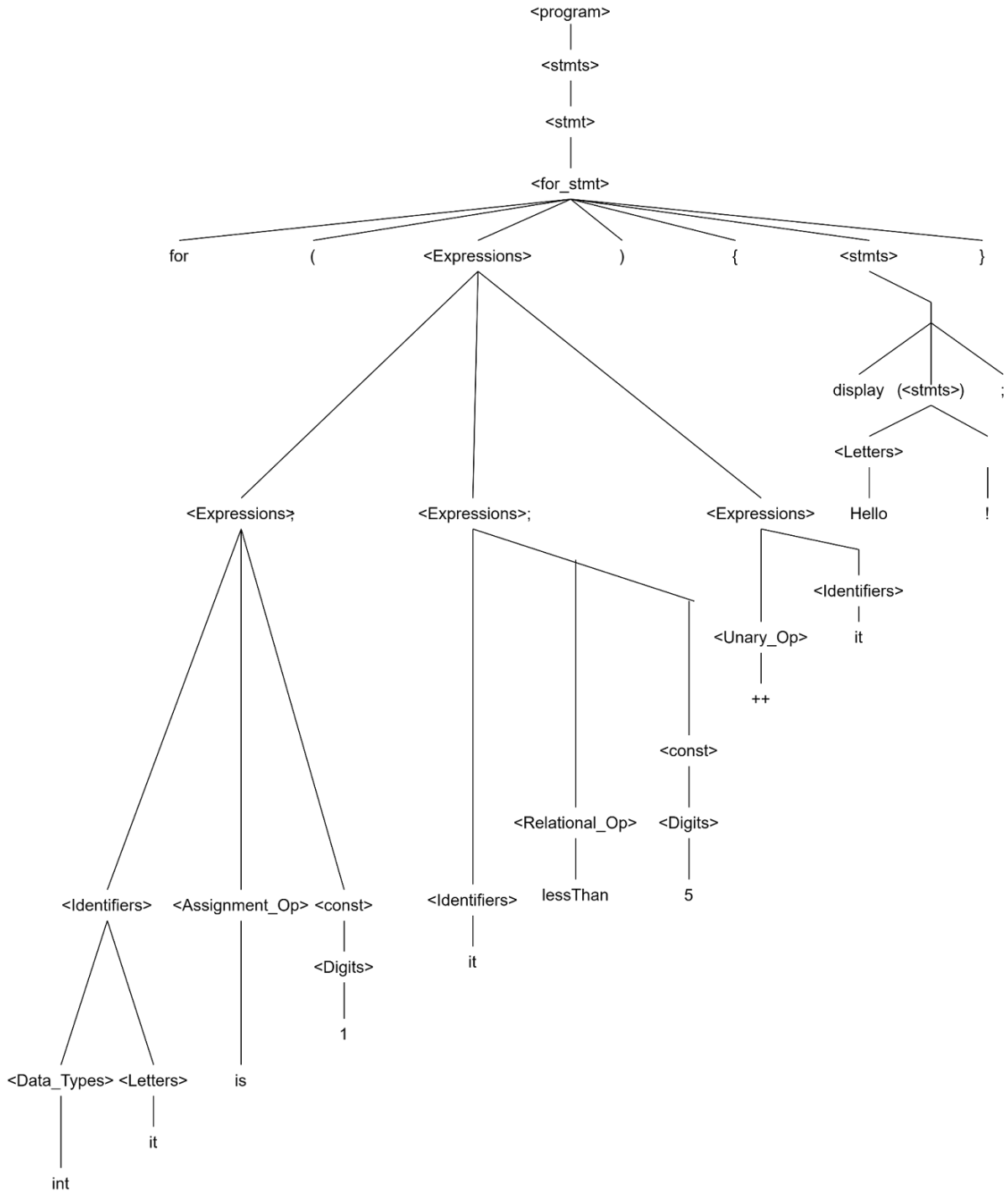
=> for (<Identifiers> <Assignment_Op> 1; it lessThan 5; ++it) {display
(“Hello!”);}
=> for (<Identifiers> is 1; it lessThan 5; ++it) {display (“Hello!”);}
=> for (<Data_Types> <Letters> is 1; it lessThan 5; ++it) {display
(“Hello!”);}
=> for (<Data_Types> it is 1; it lessThan 5; ++it) {display (“Hello!”);}
=> for (int it is 1; it lessThan 5; ++it) {display (“Hello!”);}

b. Parse tree

Leftmost (Top-Down - Left-Right)


<program>
<stmts>
<stmt>
<for_stmt>
for (<Expressions>) { <stmts> }
<Expressions> ; <Expressions> ; <Expressions>
<Identifiers> <Assignment_Op> <const>
<Data_Types> <Letters>
int
it
is
<Digits>
1
<Identifiers>
it
<Relational_Op>
lessThan
<const>
<Digits>
5
<Unary_Op>
++
<Identifiers>
it
display (<stmts>) ;
<Letters> <Special Characters>
Hello
!

Rightmost (Top-Down - Right-Left)



- **Nested for Condition**

Grammar Rule

<nested_for_stmt> ::== for (<Expressions>*) {for_stmt^{+} {<stmts>+}}

Example

```
for (int it=0; it<5; it++){
        for (int lt = 0; lt<3; lt++){
                display("Hello!");
        }
}
```

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <nested_for_stmt>
<nested_for_stmt> => for (<Expressions>*) {<for_smt>{<stmts>}}
=> for (<Expressions>; <Expressions>; <Expressions>)
{<for_smt>{<stmts>}}
=> for (<Identifiers> <Assignment_Op> <const>; <Expressions>;
<Expressions>) {<for_smt>{<stmts>}}
=> for (<Data_Types> <Letters> <Assignment_Op> <const>;
<Expressions>; <Expressions>) {<for_smt>{<stmts>}}
=> for (int <Letters> <Assignment_Op> <const>; <Expressions>;
<Expressions>) {<for_smt>{<stmts>}}
=> for (int it <Assignment_Op> <const>; <Expressions>;
<Expressions>) {<for_smt>{<stmts>}}
=> for (int it = <const>; <Expressions>; <Expressions>)
{<for_smt>{<stmts>}}
=> for (int it = <Digits>; <Expressions>; <Expressions>)
{<for_smt>{<stmts>}}
=> for (int it = 0; <Expressions>; <Expressions>)
{<for_smt>{<stmts>}}
=> for (int it = 0; <Identifiers> <Relational_Op> <const>;
<Expressions>) {<for_smt>{<stmts>}}
=> for (int it = 0; it <Relational_Op> <const>; <Expressions>)
{<for_smt>{<stmts>}}
=> for (int it = 0; it lessThan <const>; <Expressions>)
{<for_smt>{<stmts>}}
=> for (int it = 0; it lessThan <Digits>; <Expressions>)
{<for_smt>{<stmts>}}
=> for (int it = 0; it lessThan 5; <Expressions>)
{<for_smt>{<stmts>}}

=> for (int it = 0; it lessThan 5; <Identifiers> <Unary_Op>)
{<for_smt>{<stmts>}}
=> for (int it = 0; it lessThan 5; it <Unary_Op>)
{<for_smt>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {<for_smt>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for <Expressions>;
<Expressions>; <Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for <Identifiers>
<Assignment_Op> <const>; <Expressions>;
<Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for <Data_Types>
<Identifiers> <Assignment_Op> <const>;; <Expressions>;
<Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int <Identifiers>
<Assignment_Op> <const>;<Expressions>;
<Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt <Assignment_Op>
<const>; <Expressions>; <Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = <const>;
<Expressions>; <Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = <Digits>;
<Expressions>; <Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0;
<Expressions>; <Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; <Identifiers>
<Relational_Op> <const>; <Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; lt
<Relational_Op> <const>; <Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; lt <const>;
<Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; lt <Digits>;
<Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; lt<3;
<Espressions>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; lt<3;
<Identifiers> <Unary_Op>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; lt<3; lt
<Unary_Op>{<stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; lt<3; lt ++
{<stmts>}}

=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; lt<3; lt ++
{display <stmts>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; lt<3; lt++
{display <Characters>}}
=> for (int it = 0; it lessThan 5; it++) {for int lt = 0; lt<3; lt++
{display (“Hello!”);}}

Rightmost (Top-Down - Right-Left)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <nested_for_stmt>
<nested_for_stmt> => for (<Expressions>*) {<for_smt>{<stmts>}}
=> for (<Expressions>*) {<for_stmt>{display <stmts>}}
=> for (<Expressions>*) {<for_stmt> {display <Characters>;}}
=> for (<Expressions>*) {<for_stmt> {display (“Hello!”);}}
=> for (<Expressions>*) {for (<Espressions>*) {display
(“Hello!”);}}
=> for (<Expressions>*) {for <Expressions>; <Expressions>;
<Expressions> {display (“Hello!”);}}
=> for (<Expressions>*) {for <Expressions>; <Expressions>;
<Identifiers> <Unary_Op> {display (“Hello!”);}}
=> for (<Expressions>*) {for <Expressions>; <Expressions>;
<Identifiers>++) {display (“Hello!”);}}
=> for (<Expressions>*) {for <Expressions>; <Expressions>;
lt++) {display (“Hello!”);}}
=> for (<Expressions>*) {for <Expressions>; <Identifiers>
<Relational_Op> <const>; lt++) {display (“Hello!”);}}
=> for (<Expressions>*) {for <Expressions>; <Identifiers>
<Relational_Op> <Digits>; lt++) {display (“Hello!”);}}
=> for (<Expressions>*) {for <Expressions>; <Identifiers>
<Relational_Op> 3; lt++) {display (“Hello!”);}}
=> for (<Expressions>*) {for <Expressions>; <Identifiers>
<3; lt++) {display (“Hello!”);}}
=> for (<Expressions>*) {for <Expressions>; lt<3; lt++) {display
(“Hello!”);}}
=> for (<Expressions>*) {for <Identifiers> <Assignment_Op>
<consts>; lt<3; lt++) {display (“Hello!”);}}
=> for (<Expressions>*) {for <Identifiers> <Assignment_Op>
<Digits>; lt<3; lt++) {display (“Hello!”);}}
=> for (<Expressions>*) {for <Identifiers> <Assignment_Op>
0; lt<3; lt++) {display (“Hello!”);}}

=> for (<Expressions>*) {for <Identifiers> =0; lt<3; lt++) {display
(“Hello!”);}}
=> for (<Expressions>*) {for <Data_Types> <Identifiers> =0;
lt<3; lt++) {display (“Hello!”);}}
=> for (<Expressions>*) {for <Data_Types> it =0; lt<3;
lt++) {display (“Hello!”);}}
=> for (<Expressions>*) {for (int lt =0; lt<3; lt++) {display
(“Hello!”);}}
=> for (<Expressions>; <Expressions>; <Expression>) {for (int lt
=0; lt<3; lt++) {display (“Hello!”);}}
=> for (<Expressions>; <Expressions>; <Identifiers>
<Unary_Op>) {for (int lt =0; lt<3; lt++) {display
(“Hello!”);}}
=> for (<Expressions>; <Expressions>; <Identifiers> ++) {for (int
lt =0; lt<3; lt++) {display (“Hello!”);}}
=> for (<Expressions>; <Expressions>; it++) {for (int lt =0; tl<3;
lt++) {display (“Hello!”);}}
=> for (<Expressions>; <Identifiers> <Relational_Op> <const>;
it++) {for (int lt =0; lt<3; lt++) {display (“Hello!”);}}
=> for (<Expressions>; <Identifiers> <Relational_Op> <Digits>;
it++) {for (int lt =0; lt<3; lt++) {display (“Hello!”);}}
=> for (<Expressions>; <Identifiers> <Relational_Op> 5;
it++) {for (int lt =0; lt<3; lt++) {display (“Hello!”);}}
=> for (<Expressions>; <Identifiers> <5; it++) {for (int lt =0; lt<3;
lt++) {display (“Hello!”);}}
=> for (<Expressions>; it<5; it++) {for (int lt =0; lt<3; lt++)
{display (“Hello!”);}}
=> for (<Identifiers> <Assignment_Op> <const>; it<5; it++) {for
(int lt =0; lt<3; lt++) {display (“Hello!”);}}
=> for (<Identifiers> <Assignment_Op> <Digits>; it<5; it++) {for
(int lt =0; lt<3; lt++) {display (“Hello!”);}}
=> for (<Identifiers> <Assignment_Op> 0; it<5; it++) {for
(int lt =0; lt<3; lt++) {display (“Hello!”);}}
=> for (<Identifiers> = 0; it<5; it++) {for (int lt =0; lt<3; lt++)
{display (“Hello!”);}}
=> for (<Data_Types> <Identifiers> = 0; it<5; it++) {for (int lt =0;
lt<3; lt++) {display (“Hello!”);}}
=> for (<Data_Types> it = 0; it<5; it++) {for (int lt =0;
lt<3; lt++) {display (“Hello!”);}}
=> for (int it = 0; it<5; it++) {for (int lt =0; lt<3; lt++) {display

(“Hello!”);}}

b. Parse tree

Leftmost (Top-Down - Left-Right)

Rightmost (Top-Down - Right-Left)

- **While Statement**

Grammar Rule

<while_stmt> ::== while (<Expressions>) {<stmt>}

Example

```
while (it !=0){
        display(“Hello!”);
}
```

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <while_stmt>
<while_stmt> => while (<Expressions>) {<stmt>}
=> while (<Identifiers> <Relational_Op> <const>) {<stmt>}
=> while (it <Relational_Op> <const>) {<stmt>}
=> while (it != <const>) {<stmt>}
=> while (it != <Digits>) {<stmt>}
=> while (it != 0) {<stmt>}
=> while (it != 0) { display <stmt>;}
=> while (it != 0) { display <Characters>;}
=> while (it != 0) { display (“Hello!”);}

Rightmost (Top-Down - Right-Left)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <while_stmt>
<while_stmt> => while (<Expressions>) {<stmt>}
=> while (<Expressions>) {display <Characters>;}
=> while (<Expressions>) {display (“Hello!”);}
=> while (<Identifiers> <Relational_Op> <const>) {display (“Hello!”);}

=> while (<Identifiers> <Relational_Op> <Digits>) {display (“Hello!”);}
=> while (<Identifiers> <Relational_Op> 0) {display (“Hello!”);}
=> while (<Identifiers> != 0) {display (“Hello!”);}
=> while (it != 0) {display (“Hello!”);}

b. Parse tree

Leftmost (Top-Down - Left-Right)

Rightmost (Top-Down - Right-Left)

- **Nested While Statement**

Grammar Rule

<nested_while_stmt> ::== while (<expressions>) {<while_stmt>$^{+}$
{<Expressions>} <Expressions>}

Example

```
while (it != 0){
        while (l !=0){
                display(“Hello!”);
        }
}
```

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <nested_while_stmt>
<nested_while_stmt> => while (<Expressions>) {<while_stmt>$^{+}$ {<stmt>}}
=> while (<Identifers> <Relational_Op> <const>){<while_stmt>$^{+}$
<stmt>}}
=> while (it <Relational_Op> <const>){<while_stmt>$^{+}$ {<stmt>}}
=> while (it != <const>){<while_stmt>$^{+}$ {<stmt>}}
=> while (it != <Digits>){<while_stmt>$^{+}$ {<stmt>}}
=> while (it != 0){<while_stmt>$^{+}$ {<stmt>}}
=> while (it != 0){ while (Expressions)* {<stmt>}}
=> while (it != 0){ while (<Identifers> <Relational_Op> <const>)
{<stmt>}}
=> while (it != 0){ while (it <Relational_Op> <const>) {<stmt>}}
=> while (it != 0){ while (it != <const>) {<stmt>}}
=> while (it != 0){ while (it != <Digits>) {<stmt>}}

=> while (it != 0){ while (it !=0) {<stmt>}}
=> while (it != 0){ while (it !=0) {display <stmt>;}}
=> while (it != 0){ while (it !=0) {display <Characters>;}}
=> while (it != 0){ while (it !=0) {display (“Hello!”);}}

Rightmost (Top-Down - Right-Left)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <nested_while_stmt>
<nested_while_stmt> => while (<Expressions>) {<while_stmt>$^{+}$ {<stmt>}}
=> while (<Expressions>) {<while_stmt>$^{+}$ {display <stmt>;}}
=> while (<Expressions>) {<while_stmt>$^{+}$ {display <Characters>;}}
=> while (<Expressions>) {<while_stmt>$^{+}$ {display (“Hello!”);}}
=> while (<Expressions>) {while (Expressions)***** {display (“Hello!”);}}
=> while (<Expressions>) {while (<Identifiers> <Relational_Op> <const>) {display (“Hello!”);}}
=> while (<Expressions>) {while (<Identifiers> <Relational_Op> <Digits>) {display (“Hello!”);}}
=> while (<Expressions>) {while (<Identifiers> <Relational_Op> 0) {display (“Hello!”);}}
=> while (<Expressions>) {while (<Identifiers> !=0) {display (“Hello!”);}}
=> while (<Expressions>) {while (lt !=0) {display (“Hello!”);}}
=> while (<Identifers> <Relational_Op> <const>) {while (lt !=0) {display (“Hello!”);}}
=> while (<Identifers> <Relational_Op> <Digits>) {while (lt !=0) {display (“Hello!”);}}
=> while (<Identifers> <Relational_Op> 0) {while (lt !=0) {display (“Hello!”);}}
=> while (<Identifers> != 0) {while (lt !=0) {display (“Hello!”);}}
=> while (it!=0) {while (lt !=0) {display (“Hello!”);}}

b. Parse tree

Leftmost (Top-Down - Left-Right)

Rightmost (Top-Down - Right-Left)

E. Assignment Statement

- **Assignment by Value**

  Grammar Rule
  <Assignment_stmt**>** ::== <Identifiers> <Asssignment_op> (<Letters> | <Digits>) ;

Example
num = 2;

c. Instantaneous Description

Leftmost (Top-Down - Left-Right)

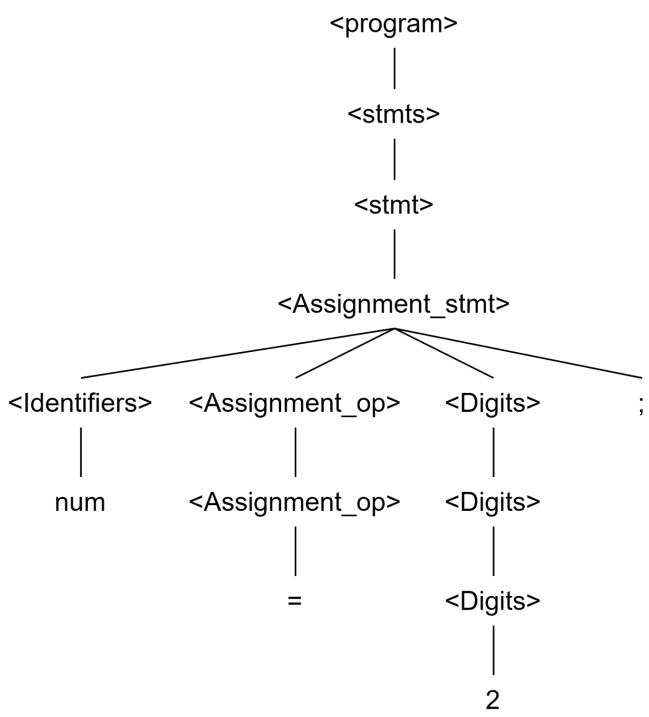
<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Assignmnet_stmt>
<Assignemt_stmt> => <Identifiers> <Assignment_Op> <Digits>;
=> num <Assignment_Op> <Digit>;
=> num = <Digit>;
=> num = 2;

Rightmost (Top-Down - Right-Left)

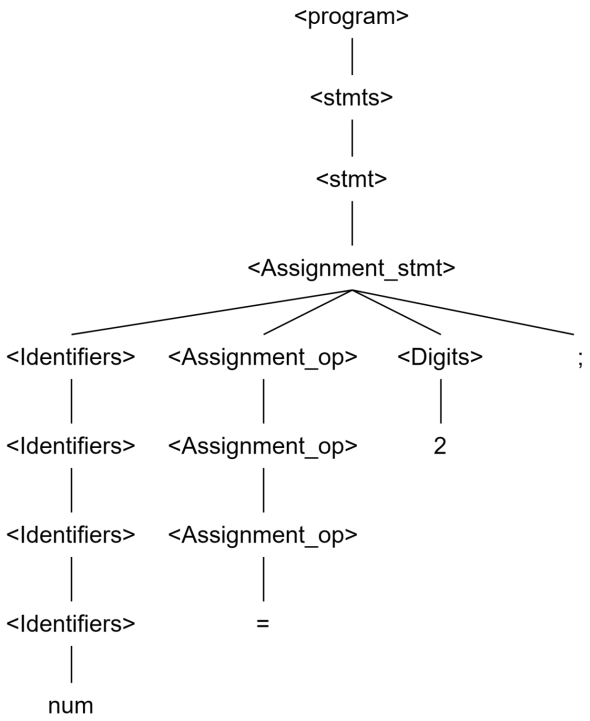
<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Assignmnet_stmt>
<Assignemt_stmts> => <Identifiers> <Assignment_Op> <Digits>;
=> <Identifiers> <Assignment_Op> 2;
=> <Identifiers> = 2;
=> num = 2;

b. Parse tree

Leftmost (Top-Down - Left-Right)



Rightmost (Top-Down - Right-Left)



- **Assignment by Variable**

Grammar Rule

<Assignment_stmt> ::== <Identifiers> <Assignment_Op> <Identifiers>;

Example

num *= num1;

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

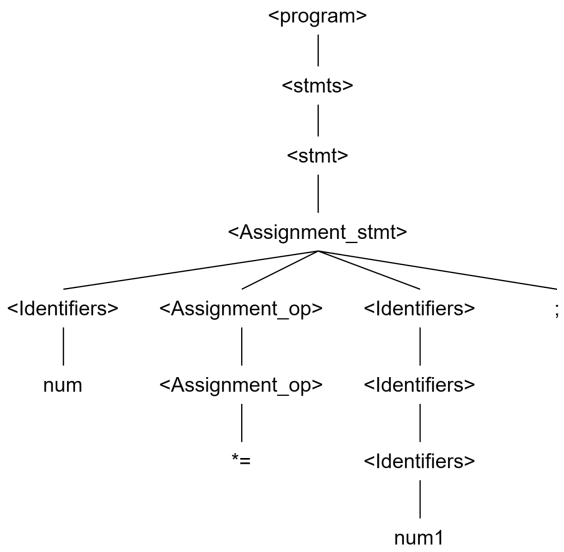
<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Assignment_stmt>
<Assignment_stmt> => <Identifiers> <Assignment_Op> <Identifiers>;
=> num <Assignment_Op> <Identifiers>;
=> num *= <Identifiers>;
=> num *= num1;

Rightmost (Top-Down - Right-Left)

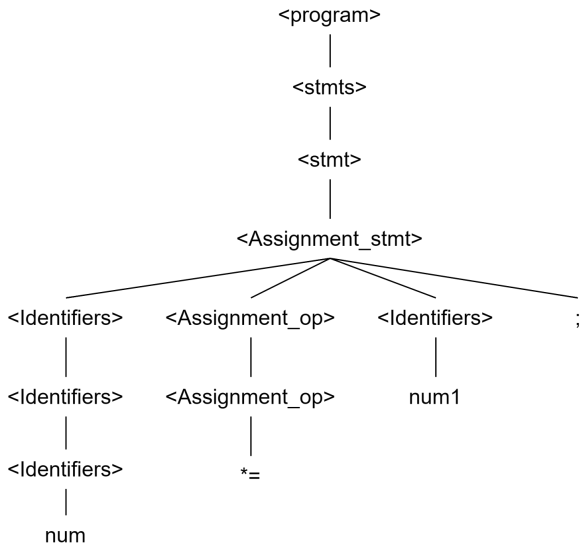
<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Assignment_stmt>
<Assignemt_stmt> => <Identifiers> <Assignment_Op> <Identifiers>;
=> <Identifiers> <Assignment_Op> num1;
=> <Identifiers> *= num1;
=> num *= num1;

b. Parse tree

Leftmost (Top-Down - Left-Right)



Rightmost (Top-Down - Right-Left)



- **Assignment by Expression**

Grammar Rule

<Assignmnet_stmt> ::== <Identifiers> <Assignment_Op> <Expressions>

Example

sum = num1 + num2 / 3;

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

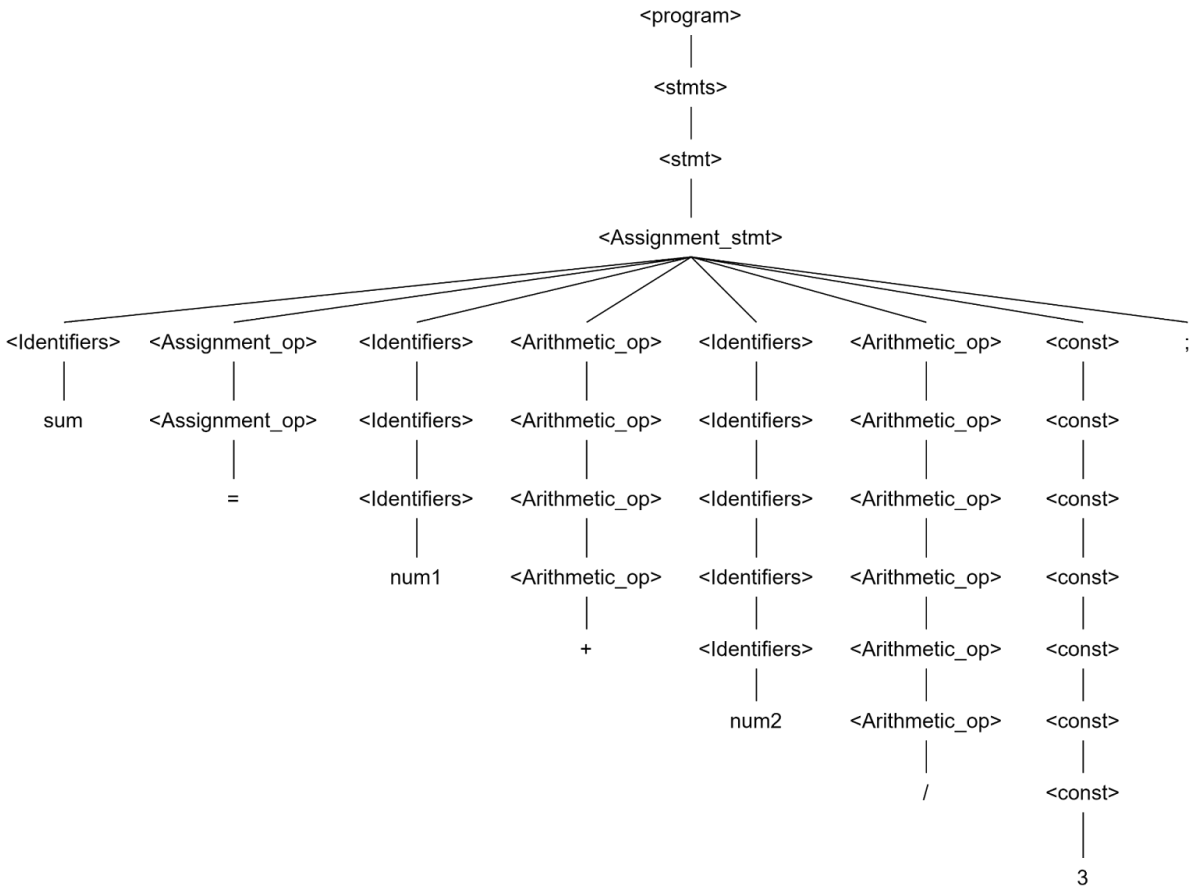
<program> => <smts>
<stmts> => <stmt>
<stmt> => <Assignment_stmts>
<Assignemt_stmts> => <Identifiers> <Assignment_Op> <Identifiers>
<Arithmetic_Op> <Identifiers> <Arithmetic_Op> <const>;
=> sum <Assignment_Op> <Identifiers>
<Arithmetic_Op> <Identifiers> <Arithmetic_Op> <const>;
=> sum = <Identifiers> <Arithmetic_Op> <Identifiers>
<Arithmetic_Op> <const>;
=> sum = num1 <Arithmetic_Op> <Identifiers>
<Arithmetic_Op> <const>;
=> sum = num1 + <Identifiers> <Arithmetic_Op> <const>;
=> sum = num1 + num2 <Arithmetic_Op> <const>;
=> sum = num1 + num2 / <const>;
=> sum = num1 + num2 / 3;

Rightmost (Top-Down - Right-Left)

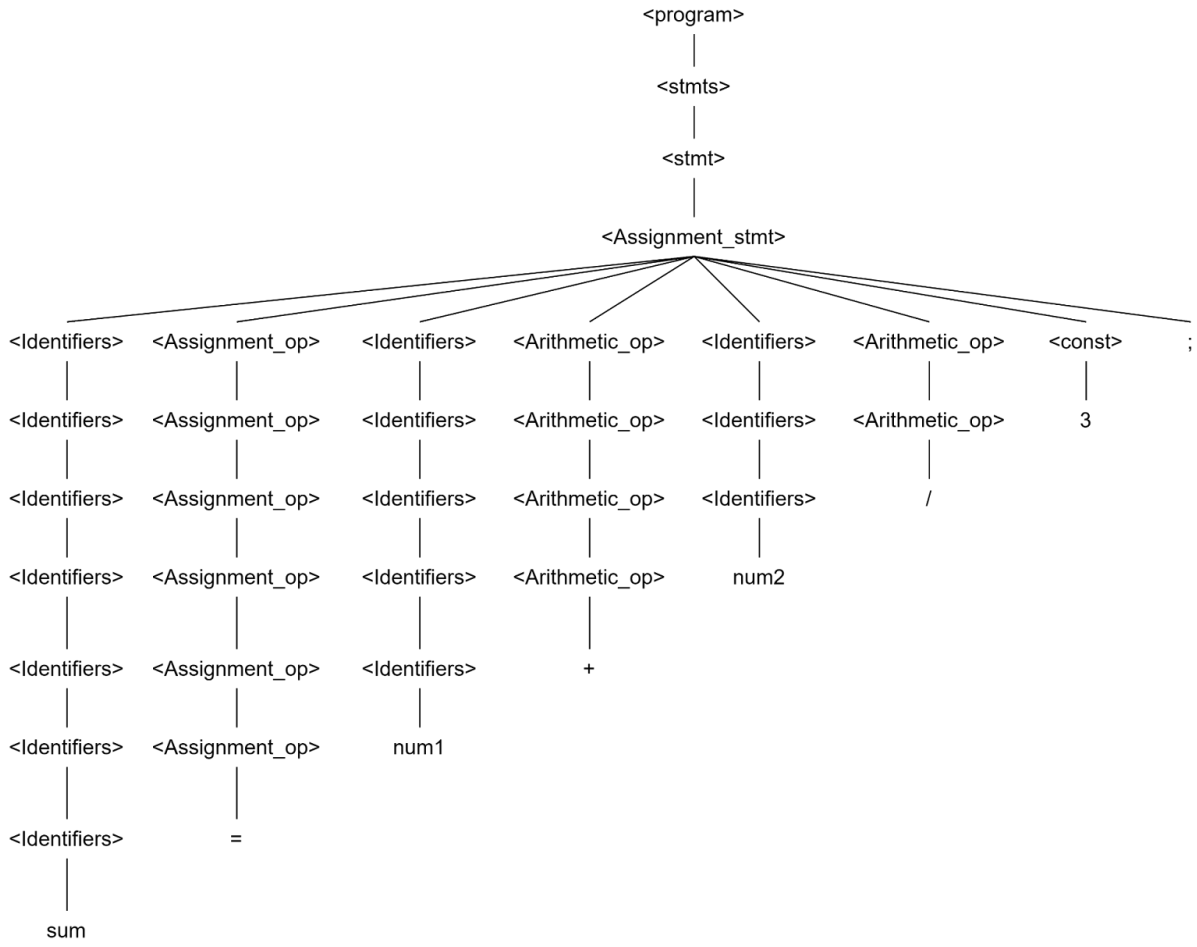
<program> => <smts>
<stmts> => <stmt>
<stmt> => <Assignment_stmts>
<Assignemt_stmt> => <Identifiers> <Assignment_Op> <Identifiers>
<Arithmetic_Op> <Identifiers> <Arithmetic_Op> <const>;
=> <Identifiers> <Assignment_Op> <Identifiers>
<Arithmetic_Op> <Identifiers> <Arithmetic_Op> 3;
=> <Identifiers> <Assignment_Op> <Identifiers>
<Arithmetic_Op> <Identifiers> / 3;
=> <Identifiers> <Assignment_Op> <Identifiers>
<Arithmetic_Op> num2 / 3;
=> <Identifiers> <Assignment_Op> <Identifiers> + num2 / 3;
=> <Identifiers> <Assignment_Op> num1 + num2 / 3;
=> <Identifiers> = num1 + num2 / 3;
=> sum = num1 + num2 / 3;

b. Parse tree

Leftmost (Top-Down - Left-Right)



Rightmost (Top-Down - Right-Left)



F. Declaration Statement

- **Declaration with Identifier**

  Grammar Rule
  <Dec_stmts**>** ::== <Data_Types> <Identifiers>;

  Example
  int num;

a. Instantaneous Description
Leftmost (Top-Down - Left-Right)
<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Dec_stmts>
<Dec_stmts> => <Data_Types> <Identifiers>;

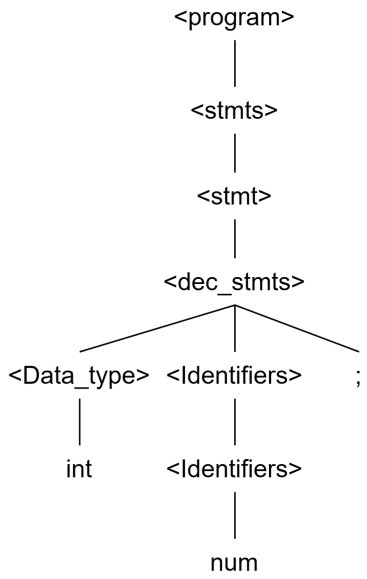
=>. int <Identifiers>;
=> int num;

Rightmost (Top-Down - Right-Left)

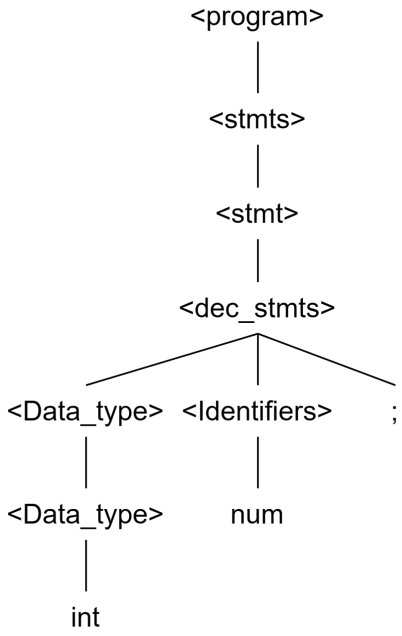
<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Dec_stmts>
<Dec_stmts> => <Data_Type> <Identifiers>;
=>. <Data_Type> num;
=> int num;

b. Parse tree

Leftmost (Top-Down - Left-Right)



Rightmost (Top-Down - Right-Left)



- **Constant Declaration**
  Grammar Rule
  <dec_stmts> ::== <Identifiers> <const>;

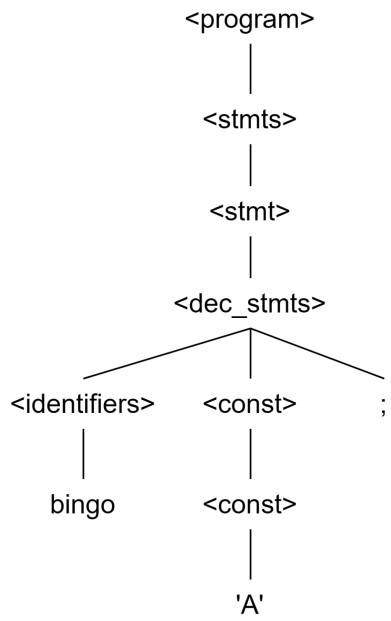
  Example
  bingo ‘A’;

a. Instantaneous Description
   Leftmost (Top-Down - Left-Right)
   <program> => <stmts>
   <stmts> => <stmt>
   <stmt> => <Dec_stmts>
   <Dec_stmts> => <identifier> <const>;
   => <bingo <const>;
   => <bingo ‘A’;

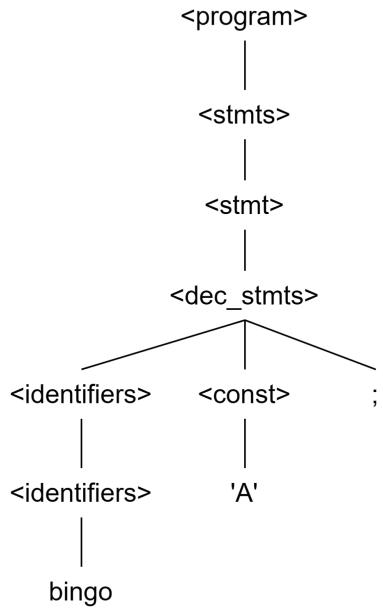
   Rightmost (Top-Down - Right-Left)
   <program> => <stmts>
   <stmts> => <stmt>
   <stmt> => <Dec_stmts>
   <Dec_stmts> => <identifier> <const>;
   => <identifier>‘A’;
   => <bingo ‘A’;

b. Parse tree

Leftmost (Top-Down - Left-Right)



Rightmost (Top-Down - Right-Left)



- **Declaring Identifier with an identifier**

  Grammar Rule

  <dec_stmts**>** **=>** <Data_Types> <Identifiers> <Assignment_Op> <Identifiers>

  Example

  int age is birthYear;

a. Instantaneous Description

Leftmost (Top-Down - Left-Right)

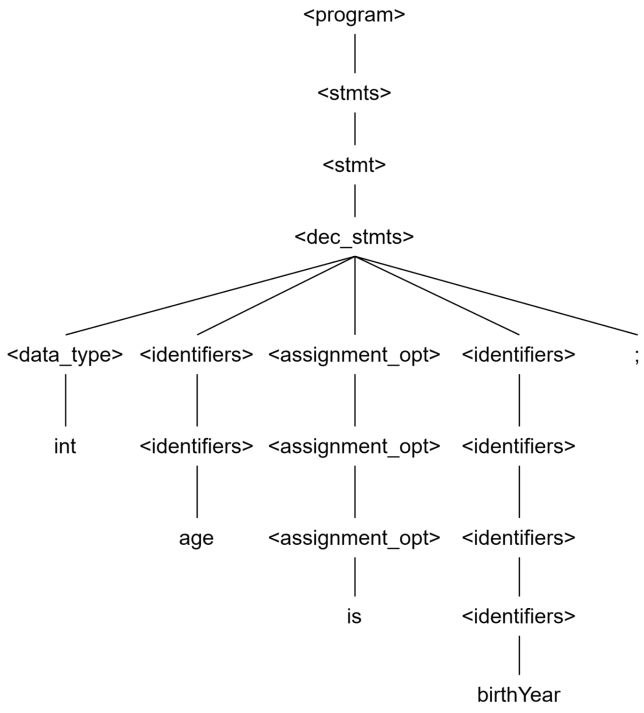
<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Dec_stmts>
<Dec_stmts> => <Data_types> <Identifiers> <Assignmnet_Op> <Identifiers>;
=> int <Identifiers> <Assignmnet_Op> <Identifiers>;
=> int age <Assignmnet_Op> <Identifiers>;
=> int age is <Identifiers>;
=> int age is birthYear;

Rightmost (Top-Down - Right-Left)

<program> => <stmts>
<stmts> => <stmt>
<stmt> => <Dec_stmts>
<Dec_stmts> => <data_type> <Identifiers> <Assignmnet_Op> <Identifiers>;
=> <data_type> <Identifiers> <Assignmnet_Op> birthYear;
=> <data_type> <Identifiers> is birthYear;
=> <data_type> age is birthYear;
=> int age is birthYear;

b. Parse tree

Leftmost (Top-Down - Left-Right)



Rightmost (Top-Down - Right-Left)