

# BNF DESCRIPTION

**<program>** ::= START<stmts>FINISH

**<stmts>** ::= <stmts><stmt> | <stmt>

**<stmt>** ::= <matched\_stmt> | <unmatched\_stmt>

**<matched\_stmt>** ::= if <LP> <logic\_exp> <RP> <matched\_stmt> else <matched> |  
<non\_if\_stmt>

**<unmatched\_stmt>** ::= if <LP> <logic\_exp> <RP> <stmt> | if <LP> <logic\_exp> <RP>  
<matched> else <unmatched>

**<non\_if\_stmt>** ::= <loop\_stmt> | <assign\_stmt> | <var\_declaration> | <comment> |  
<func\_call\_stmt> | <func\_declaration> | <return\_stmt>

**<assign\_stmt>** ::= <lside> <equal> <rside>

**<rside>** ::= <logical\_expression>

**<logical\_expression>** ::= <or\_expression>

**<or\_expression>** ::= <or\_expression> <or\_relation><and\_expression> | <and\_expression>

**<and\_expression>** ::= <and\_expression><and\_relation><comparison\_expression> |  
<comparison\_expression>

**<comparison\_expression>** ::=  
<comparison\_expression><comparator><add\_sub\_expression> | <add\_sub\_expression>

**<add\_sub\_expression>** ::= <add\_sub\_expression> <plus> <mul\_div\_expression> |  
<add\_sub\_expression> <minus> <mul\_div\_expression> | <mul\_div\_expression>

**<mul\_div\_expression>** ::= <mul\_div\_expression> <multiply> <pow\_expression> |  
<mul\_div\_expression> <divide> <pow\_expression> | <pow\_expression>

**<pow\_expression>** ::= <pow\_expression> <pow\_sign> <element> | <element>

**<element>** ::= <LP> <logical\_expression> <RP> | <var\_id> | <integer> | <double\_value> |  
<array\_init> | <string\_init> | <connect\_init>

**<lside> ::= <var\_id> | <var\_declaration>**

**<comment> ::= <comment\_string>**

**<code\_block> ::= // <stmts> \\\**

**<string\_init> ::= <string>**

## ARRAYS

**<array\_init> ::= <LSB><array\_elements><RSB>**

**<array\_elements> ::= <array\_elements> <comma> <array\_element> | <array\_element>**

**<array\_element> ::= <integer> | <double\_value> | <string>**

## DECLARATIONS

**<var\_declaration> ::= <type\_id> <var\_id> | <type\_id> array <var\_id>**

**<type\_id> ::= <int> | <double> | <string\_type> | <connection>**

**<var\_id> ::= <identifier>**

**<func\_name> ::= <identifier>**

**<int> ::= int**

**<double> ::= double**

**<string\_type> ::= string**

**<array> ::= array**

**<connection> ::= connection**

## LOOPS

**<loop\_stmt> ::= <for\_stmt> | <while\_stmt>**

**<for\_stmt> ::= for <LP> <var\_id> in <var\_id> <RP> <codeblock> | for <LP> <assign\_stmt>  
where <logical\_expression> with <assign\_stmt> <RP> <codeblock> | for <LP> <assign\_stmt>  
where <logical\_expression> <RP> <codeblock>**

**<while\_stmt>** ::= while <LP> <logical\_expression> <RP> <code\_block>

## FUNCTIONS

**<func\_call\_stmt>** ::= <primitive\_func\_call> | <non\_primitive\_func\_call>

**<primitive\_func\_call>** ::= <read\_temp> | <read\_hum> | <read\_air\_p> | <read\_air\_q> |  
<read\_light> | <read\_sound\_lvl> | <read\_timestamp\_from\_timer> |  
<send\_integer\_to\_connection> | <read\_integer\_from\_connection> | <connect>

**<non\_primitive\_func\_call>** ::= <func\_name> <LP> <parameter\_list\_on\_call>? <RP>

**<parameter\_list\_on\_call>** ::= <rside> | <parameter\_list\_on\_call> <comma> <rside>

**<func\_declaration>** ::= func <func\_name> <LP> <parameter\_list>? <RP>

**<return\_stmt>** ::= return <var\_id> | return <double\_value> | return <integer> | return <string>

**<parameter\_list>** ::= <var\_declaration> | <parameter\_list> <comma> <var\_declaration>

**<read\_temp>** ::= read\_temp<LP><RP>

**<read\_hum>** ::= read\_humidity<LP><RP>

**<read\_air\_p>** ::= read\_air\_pressure<LP><RP>

**<read\_air\_q>** ::= read\_air\_quality<LP><RP>

**<read\_light>** ::= read\_light<LP><RP>

**<read\_sound\_lvl>** ::= read\_sound\_level<LP><double\_value><RP> |  
read\_sound\_level<LP><var\_id><RP>

**<read\_timestamp\_from\_timer>** ::= read\_timer<LP><RP>

**<send\_integer\_to\_connection>** ::= send\_int<LP><var\_id><comma><var\_id><RP> |  
send\_int<LP><var\_id><comma><integer><RP>

**<read\_integer\_from\_connection>** ::= read\_int<LP><var\_id><RP>

**<connect\_init>** ::= connect<LP><var\_id><RP> | connect<LP><string><RP>

## SWITCHES

**<control\_switches>** ::= <switch\_state>

**<switch\_states>** ::= <turn\_on> | <turn\_off>

**<turn\_on>** ::= SwitchON1 | SwitchON2 | SwitchON3 | SwitchON4 | SwitchON5 | SwitchON6 | SwitchON7 | SwitchON8 | SwitchON9 | SwitchON10

**<turn\_off>** ::= SwitchOFF1 | SwitchOFF2 | SwitchOFF3 | SwitchOFF4 | SwitchOFF5 | SwitchOFF6 | SwitchOFF7 | SwitchOFF8 | SwitchOFF9 | SwitchOFF10

## SYMBOLS

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

**<integer>** ::= <digit> | <integer> <digit>

**<double\_value>** ::= <integer><dot><integer>

**<string>** ::= "<string\_characters>"

**<comment\_string>** ::= #<string\_characters>#

**<string\_characters>** ::= <character> | <string\_characters> <character>

**<character>** ::= <lowercase\_letter> | <uppercase\_letter> | <digit> | <symbols> | <space>

**<lowercase\_letter>** ::= 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

**<uppercase\_letter>** ::= 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

**<symbols>** ::= ! | ' | ^ | # | + | \$ | % | & | / | { | ( | [ | ) | ] | = | ? | \* | \ | - | " | | | : | . |

**<space>** ::= SPACE

## TOKENS

**<identifier>** ::= <lowercase\_letter> | <identifier><lowercase\_letter> | <identifier><underscore> |  
<identifier><digit>

**<and>** ::= &

**<and\_relation>** ::= <and><and>

**<or>** ::= |

**<or\_relation>** ::= <or><or>

**<equal>** ::= =

**<comma>** ::= ,

**<LP>** ::= (

**<RP>** ::= )

**<LSB>** ::= [

**<RSB>** ::= ]

**<minus>** ::= -

**<underscore>** ::= \_

**<plus>** ::= +

**<multiply>** ::= \*

**<divide>** ::= /

**<hashtag>** ::= #

**<pow\_sign>** ::= ^

**<comparator>** ::= > | < | >= | <= | ==