BL Context-Free Grammar

Program Parse

program → **PROGRAM** *identifier* **IS**
{ *instruction* }
BEGIN
block
END *identifier*

instruction → **INSTRUCTION** *identifier* **IS**
block
END *identifier*

Statement Parse

block → { *statement* }

statement → *if* | *while* | *call*

if → **IF** *condition* **THEN**
block
[**ELSE**
block]
END IF

while → **WHILE** *condition* **DO**
block
END WHILE

call → *identifier*

Tokenizer

identifier → *letter* { *letter* | *digit* | **-** }

letter → `a` | `b` | ... | `z` | `A` | `B` | ... | `Z`

digit → `0` | `1` | ... | `9`

keyword → `BEGIN` | `DO` | `ELSE` | `END` |
`IF` | `INSTRUCTION` | `IS` |
`PROGRAM` | `THEN` | `WHILE`

condition → `next-is-empty` | `next-is-not-empty` |
`next-is-wall` | `next-is-not-wall` |
`next-is-enemy` | `next-is-not-enemy` |
`next-is-friend` | `next-is-not-friend` |
`random` | `true`

Note 1: The special symbols { and } mean that the enclosed sequence of symbols occurs *zero or more times*, and the special symbols [and] mean that the enclosed sequence of symbols occurs *zero or one times*.

Note 2: Even though the rewrite rules for *identifier* do not rule this out explicitly, keywords and conditions are ***not*** considered identifiers (as defined by the mathematical subtype IDENTIFIER in the StatementKernel interface).