Model language

<program> → <global_declarations> **main** <main_arguments> <body>
<global_declarations> → {<global_declaration>}
<global_declaration> → **func** <function> | **array of** <array> | <variable>
<array> → <type> <array_size> <id> {**,** <id>}
<variable> → <type> <id> [<init>] {**,** <id> [<init>]}
<declaration> → **array of** <array> | <variable>
<array_size> → **[** {<num> **,** } <num> **]**
<init> → = <constant>
<type> → **int** | **real** | **bool** | **str** | **char**
<function> → <type> <id> <arguments> <body>
<main_arguments> → **(** [<type> <id>] {, <type> <id>} **)**
<arguments> → **(** [<type> <id>] {, <type> <id>} **)**
<body> → **{** {body_operator} **}**
<block> → **;** | **{** <operator_> **}** | <operator_>
<body_operator> → <declaration> | <operator_>
<operator_> → **;** | **if** <operator_if> | **while** <operator_while> |
do <operator_do> | **for** <operator_for> | **switch** <operator_switch> |
read <operator_read> | **write** <operator_write> |
break<operator_break> | **continue** <operator_continue> |
return <operator_return> | <operator_simple>

<operator_simple> → <mono1> **;** | = <expression>

<operator_if> → **(** <expression> **)** <block> [**else** <block>] |
<operator_while> → **(** <expression> **)** <block> |
<operator_do> → <block> **while** **(** <expression> **)** |
<operator_for> → **(**[<expression>]**;** [<expression>]**;** [<expression>]**)** <block>
<operator_read> → **(**<identificator> {, < identificator> } **)**
<operator_write> → **(**<expression> { , <expression>} **)**
<operator_return> → <expression>
<operator_break> →
<operator_continue> →
<index> → **[** <expression> { **,** <expression> }**]**
<expression> → <value> { = <value> }
<value> → <or_value> { **or** <or_value> }
<or_value> → <and_value> {**and** <and_value> }
<and_value> → [**not**] <bool_value> {<bool_operation> <bool_value> }
<bool_value> → <add_value> {+ | - <add_value> }
<add_value> → <mul_value> {* | / <mul_value> }
<mul_value> → [+ | -] (<expression>) | <sign>
<mono> → <mono1> | <constant>
<mono1> → <id> <call_func> | <var_ident>
<var_ident> → **[** <index> **]**
<call_func> → **(**<expression> {, <expression>}**)**