CILA

Language specification

Jakub Mendyk

May 9, 2019

1 Grammar

integer	::=	*digit* | integer *digit*
keyword	::=	if | then | else | fi | while | do | od | div | mod | or | and | not | let
alfanum	::=	*letter* | alfanum *letter* | alfanum *digit*
ident	::=	alfanum (not in keyword)
⟨program⟩	::=	⟨instruction⟩| ⟨program⟩⟨instruction⟩
⟨instruction⟩	::=	**let ident** := ⟨arith_expr⟩;
	|	**let ident**[⟨arith_expr⟩] := ⟨arith_expr⟩;
	|	**ident** := ⟨arith_expr⟩;
	|	**ident**[⟨arith_expr⟩] := { ⟨arith_expr⟩, ... } ;
	|	**if** ⟨logic_expr⟩**then** ⟨program⟩**fi**
	|	**if** ⟨logic_expr⟩**then** ⟨program⟩**else** ⟨program⟩**fi**
	|	**while** ⟨logic_expr⟩**do** ⟨program⟩**od**
⟨logic_expr⟩	::=	⟨logic_summand⟩| ⟨logic_expr⟩**or** ⟨logic_summand⟩
⟨logic_summand⟩	::=	⟨logic_multiplicand⟩| ⟨logic_summand⟩**and** ⟨logic_multiplicand⟩
⟨logic_multiplicand⟩	::=	⟨rel_expr⟩| **not** ⟨logic_multiplicand⟩
⟨rel_expr⟩	::=	⟨arith_expr⟩⟨rel_op⟩⟨arith_expr⟩| (⟨logic_expr⟩)
⟨rel_op⟩	::=	= | < | > | <= | >= | <>
⟨arith_expr⟩	::=	⟨arith_summand⟩| ⟨arith_expr⟩⟨summ_op⟩⟨arith_summand⟩
⟨arith_summand⟩	::=	⟨arith_multiplicand⟩| ⟨arith_summand⟩⟨mult_op⟩⟨arith_multiplicand⟩
⟨arith_multiplicand⟩	::=	⟨simple_expr⟩| ⟨simple_expr⟩^ ⟨arith_multiplicand⟩
⟨simple_expr⟩	::=	(⟨arith_expr⟩) | **integer** | **ident** | **ident**[⟨arith_expr⟩]
⟨summ_op⟩	::=	+ | −
⟨mult_op⟩	::=	***** | **div** | **mod**