Appendix to the VIP manual - the VIP syntax

Throughout the grammar the following abbreviations are used:

- { X } — X is optional (0 or 1 repetitions of X)
- < X > — any number of repetitions of X (possibly 0)
- (X | Y) — X or Y

1 Declarations

Program	=	**program** { *Preamble* }
		{ **type** *TypeDecls* }
		{ **var** *VarDecls* }
		{ *FunDecls* }
		begin *Instruction* **end.**
TypeDecls	=	*TypeDecl* < **;** *TypeDecl* >
TypeDecl	=	*Ident* **=** *TypeDescr*
TypeDescr	=	*Ident* | *SimpleType* | *RecordType* |
		ArrayType | *PointerType*
SimpleType	=	**integer** | **real** | **boolean**
RedordType	=	**record** *VarDecls* **end**
ArrayType	=	**array** [*ArraySizes*] **of** *TypeDescr*
ArraySizes	=	*ArraySize* < **,** *ArraySize* >
ArraySize	=	*NumExpr* **..** *NumExpr*
PointerType	=	**^** *TypeDescr*
VarDecls	=	*VarDeclLine* < **;** *VarDeclLine* >
VarDeclLine	=	*Ident* < **,** *Ident* > **:** *TypeDescr*
FunDecls	=	*FunDecl* < **;** *FunDecl* >
FunDecl	=	(**function** | **procedure**) *Preamble* { **:** *TypeDescr* } **;**
		{ **var** *VarDecls* }
		begin *Instruction* **end**
Preamble	=	*Ident* { **(** *ParamDecls* **)** } **;**
ParamDecls	=	*ParamDecl* < **;** *ParamDecl* >
ParamDecl	=	{ **var** } *VarDeclLine*

2 Expressions

Expr	=	*NumExpr* | *BoolExpr* | **nil**
NumExpr	=	*NumExpr* (+ | - | * | / | **div** | **mod**) *NumExpr* |
		(*NumExpr*) | - *NumExpr* |
		sqrt (*NumExpr*) |
		(**min** | **max**) (*NumExpr* < , *NumExpr* >) |
		AtomExpr | *Number*
BoolExpr	=	*BoolExpr* (**and** | **or**) *BoolExpr* |
		(*BoolExpr*) | **not** *BoolExpr* |
		Expr (= | <>) *Expr* |
		NumExpr (< | <= | > | =>) *NumExpr* |
		AtomExpr | **true** | **false**
AtomExpr	=	*LValue* | ^ *LValue* | *FunCall*
LValue	=	*Ident* |
		AtomExpr . *Ident* |
		AtomExpr [*NumExpr* < , *NumExpr* >] |
		AtomExpr ^ |
FunCall	=	*Ident* { ({ *Expr* < , *Expr* > }) }

3 Instructions

Instructions	=	*Instruction* < ; *Instruction* >
Instruction	=	**begin** *Instructions* **end** |
		InstrIfte | *InstrFor* | *InstrWhile* | *InstrAssign* |
		InstrWrite | *InstrSwap* | *InstrNew* | *InstrDispose* |
		FunCall
InstrIfte	=	**if** *BoolExpr* **then** *Instruction* { **else** *Instruction* }
InstrFor	=	**for** *Ident* := *NumExpr* (**to** | **downto**) *NumExpr* **do** *Instruction*
InstrWhile	=	**while** *BoolExpr* **do** *Instruction*
InstrAssign	=	*LValue* := *Expr*
InstrSwap	=	**swap** (*LValue* , *LValue*)
InstrWrite	=	(**Write** | **Writeln**) (*Writable* < , *Writable* >)
Writable	=	' *String* ' | *NumExpr*
InstrNew	=	**new** (*LValue*)
InstrDispose	=	**dispose** (*AtomExpr*)

4 Accessories

String	=	any string
Ident	=	any string consisting of letters and digits, beginning with a letter
Number	=	{ - } *Digits* { . *Digits* }
Digits	=	any string consisting of decimal digits

5 Comments

Comments in the programs are between { and } or from // till the end of line (C-style).