

# BNF for the Gryph Programming Language

Vitor Greati

Artur Curinga

Carlos Vieira

Vinícius Campos

May 14, 2018

## Contents

<b>1</b>	<b>General structure</b>	<b>1</b>
1.1	Program . . . . .	1
1.2	Statements . . . . .	1
1.2.1	IO . . . . .	1
1.2.2	Variables . . . . .	2
1.3	Subprograms . . . . .	2
1.3.1	Declaration . . . . .	2
1.3.2	Call . . . . .	2
<b>2</b>	<b>Control Structures</b>	<b>2</b>
2.1	Ifelse statements . . . . .	2
<b>3</b>	<b>Types</b>	<b>2</b>
<b>4</b>	<b>Expressions</b>	<b>2</b>
4.1	Any expression . . . . .	2
4.2	Relational expressions . . . . .	2
4.3	Boolean expressions . . . . .	3
4.4	Expressions with numbers, lists and strings . . . . .	3

## 1 General structure

### 1.1 Program

$$\begin{aligned}\langle \text{program} \rangle & \models \langle \text{program-unit} \rangle \mid \langle \text{program-unit} \rangle \langle \text{program} \rangle \\ \langle \text{program-unit} \rangle & \models \langle \text{stmt} \rangle ; \mid \langle \text{subprog-decl} \rangle\end{aligned}$$

### 1.2 Statements

$$\begin{aligned}\langle \text{stmt-list} \rangle & \models \langle \text{stmt} \rangle ; \mid \langle \text{stmt} \rangle ; \langle \text{stmt-list} \rangle \\ \langle \text{stmt} \rangle & \models \langle \text{read-stmt} \rangle \mid \langle \text{print-stmt} \rangle \mid \langle \text{var-decl-stmt} \rangle\end{aligned}$$

#### 1.2.1 IO

$$\begin{aligned}\langle \text{read-stmt} \rangle & \models \text{read } \langle \text{ident} \rangle \\ \langle \text{write-stmt} \rangle & \models \text{print } \langle \text{ident} \rangle \mid \text{print } \langle \text{string-lit} \rangle\end{aligned}$$

### 1.2.2 Variables

$$\begin{aligned}\langle \text{ident-begin-stmt} \rangle & \models \langle \text{ident-list} \rangle \langle \text{ident-list-post} \rangle \\ \langle \text{ident-list-post} \rangle & \models \langle \text{type} \rangle \langle \text{var-decl-stmt} \rangle \mid \langle \text{var-attr-stmt} \rangle \\ \langle \text{var-decl-stmt} \rangle & \models \lambda \mid \langle \text{var-attr-stmt} \rangle \\ \langle \text{var-attr-stmt} \rangle & \models = \langle \text{expr-list} \rangle\end{aligned}$$

## 1.3 Subprograms

### 1.3.1 Declaration

### 1.3.2 Call

$$\langle \text{subprog-call} \rangle \models \langle \text{ident} \rangle (\langle \text{expr-list} \rangle)$$

## 2 Control Structures

### 2.1 Ifelse statements

$$\langle \text{if-stmt} \rangle \models (\langle \text{b-expr} \rangle) \langle \text{block-or-stmt} \rangle$$

## 3 Types

$$\begin{aligned}\langle \text{type-list} \rangle & \models \langle \text{type} \rangle, \langle \text{type-list} \rangle \mid \langle \text{type} \rangle \\ \langle \text{type} \rangle & \models \langle \text{native-type} \rangle \mid \langle \text{user-type} \rangle \\ \langle \text{native-type} \rangle & \models \langle \text{primitive-type} \rangle \mid \langle \text{composite-type} \rangle \\ \langle \text{primitive-type} \rangle & \models \text{int} \mid \text{float} \mid \text{char} \mid \text{string} \\ \langle \text{composite-type} \rangle & \models [\langle \text{type} \rangle] \mid |\langle \text{type} \rangle| \mid (\langle \text{type} \rangle, \langle \text{type-list} \rangle) \mid \langle \text{graph-type} \rangle \\ \langle \text{graph-type} \rangle & \models <\langle \text{type} \rangle> \mid <\langle \text{type} \rangle, \langle \text{type} \rangle> \\ \langle \text{user-type} \rangle & \models \langle \text{upper-letter} \rangle \langle \text{alpha-num-list} \rangle\end{aligned}$$

### Observations

- The maximum size of tuples depends on the language implementation, though, in the BNF description above, it may assume any value.

## 4 Expressions

### 4.1 Any expression

$$\langle \text{any-expr} \rangle \models \langle \text{rel-expr} \rangle \mid \langle \text{bool-expr} \rangle \mid \langle \text{expr} \rangle$$

### 4.2 Relational expressions

$$\begin{aligned}\langle \text{rel-expr} \rangle & \models \langle \text{rel-term} \rangle \langle \text{rel-expr-aux} \rangle \\ \langle \text{rel-expr-aux} \rangle & \models \langle \text{rel-op} \rangle \langle \text{rel-term} \rangle \mid \langle \text{rel-op} \rangle \langle \text{rel-term} \rangle \langle \text{rel-expr-aux} \rangle \\ \langle \text{rel-op} \rangle & \models > \mid < \mid \leq \mid \geq \mid == \mid != \\ \langle \text{rel-term} \rangle & \models \langle \text{any-expr} \rangle\end{aligned}$$

### 4.3 Boolean expressions

$$\begin{aligned}\langle \text{b-expr} \rangle &\models \langle \text{b-term} \rangle \mid \langle \text{b-term} \rangle \langle \text{b-expr-aux} \rangle \\ \langle \text{b-expr-aux} \rangle &\models \langle \text{b-bin-op-p0} \rangle \langle \text{b-term} \rangle \mid \langle \text{b-bin-op-p0} \rangle \langle \text{b-term} \rangle \langle \text{b-expr-aux} \rangle \\ \langle \text{b-term} \rangle &\models \langle \text{b-un-op} \rangle \langle \text{b-term} \rangle \mid \langle \text{b-un-op} \rangle \langle \text{b-term} \rangle \langle \text{b-term-aux} \rangle \mid \langle \text{b-factor} \rangle \mid \langle \text{b-factor} \rangle \langle \text{b-term-aux} \rangle \\ \langle \text{b-term-aux} \rangle &\models \langle \text{b-bin-op-p1} \rangle \langle \text{b-factor} \rangle \mid \langle \text{b-bin-op-p1} \rangle \langle \text{b-factor} \rangle \langle \text{b-term-aux} \rangle \\ \langle \text{b-un-op} \rangle &\models \text{not} \\ \langle \text{b-bin-op-p0} \rangle &\models \text{or} \mid \text{xor} \\ \langle \text{b-bin-op-p1} \rangle &\models \text{and} \\ \langle \text{b-factor} \rangle &\models ( \langle \text{b-expr} \rangle ) \mid \text{true} \mid \text{false} \mid \langle \text{rel-expr} \rangle\end{aligned}$$

### 4.4 Expressions with numbers, lists and strings

$$\begin{aligned}\langle \text{expr} \rangle &\models \langle \text{term} \rangle \mid \langle \text{term} \rangle \langle \text{expr-aux} \rangle \\ \langle \text{expr-aux} \rangle &\models \langle \text{bin-op-p0} \rangle \langle \text{term} \rangle \mid \langle \text{bin-op-p0} \rangle \langle \text{term} \rangle \langle \text{expr-aux} \rangle \\ \langle \text{term} \rangle &\models \langle \text{un-op} \rangle \langle \text{term} \rangle \mid \langle \text{un-op} \rangle \langle \text{term} \rangle \langle \text{term-aux} \rangle \mid \langle \text{factor} \rangle \mid \langle \text{factor} \rangle \langle \text{term-aux} \rangle \\ \langle \text{term-aux} \rangle &\models \langle \text{bin-op-p1} \rangle \langle \text{factor} \rangle \mid \langle \text{bin-op-p1} \rangle \langle \text{factor} \rangle \langle \text{term-aux} \rangle \\ \langle \text{factor} \rangle &\models \langle \text{literal} \rangle ^ \langle \text{factor} \rangle \mid \langle \text{literal} \rangle \\ \langle \text{literal} \rangle &\models \langle \text{basis} \rangle \mid + \langle \text{basis} \rangle \mid - \langle \text{basis} \rangle \\ \langle \text{basis} \rangle &\models ( \langle \text{expr} \rangle ) \mid \langle \text{ident} \rangle \mid \langle \text{int-lit} \rangle \mid \langle \text{float-lit} \rangle \mid \langle \text{list-lit} \rangle \mid \langle \text{subprog-call} \rangle \mid \langle \text{string-lit} \rangle \\ \langle \text{un-op} \rangle &\models + \mid - \\ \langle \text{bin-op-p0} \rangle &\models + \mid - \\ \langle \text{bin-op-p1} \rangle &\models * \mid / \mid \% \mid ++ \mid **\end{aligned}$$