BNF for the Gryph Programming Language

Vitor Greati Artur Curinga Carlos Vieira Vinícius Campos

June 13, 2018

Contents

1 General structure

1.1 Program

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

1.2 Identifiers

$$\begin{aligned} \langle \text{id-list} \rangle &\models \langle \text{identifier} \rangle \{ \mathbf{,} \langle \text{identifier} \rangle \} \\ \langle \text{identifier} \rangle &\models \langle \text{alpha} \rangle \langle \text{id-tail} \rangle \\ \langle \text{user-type-id} \rangle &\models \langle \text{upper-alpha} \rangle \langle \text{id-tail} \rangle \\ \langle \text{id-tail} \rangle &\models \{ \langle \text{alpha-num} \rangle \} \{ \mathbf{'} \} \\ \langle \text{upper-alpha} \rangle &\models \mathbf{A} \mid \ldots \mid \mathbf{Z} \\ \langle \text{alpha} \rangle &\models \langle \text{upper-alpha} \rangle \mid (\mathbf{a} \mid \ldots \mid \mathbf{z}) \\ \langle \text{alpha-num} \rangle &\models \langle \text{alpha} \rangle \mid (\mathbf{0} \mid \ldots \mid \mathbf{9}) \mid \mathbf{\_} \end{aligned}$$

1.3 Statements

$$\begin{aligned} \langle \text{stmt-list} \rangle &\models \langle \text{stmt} \rangle \{ \langle \text{stmt} \rangle \} \\ \langle \text{stmt-block} \rangle &\models \mathbf{\{} \ \langle \text{stmt-list} \rangle \ \mathbf{\}} \\ \langle \text{block-or-matched} \rangle &\models \langle \text{stmt-block} \rangle \mid \langle \text{matched-stmt} \rangle \\ \langle \text{com-stmt} \rangle &\models (\langle \text{read-stmt} \rangle \mid \langle \text{print-stmt} \rangle \mid \langle \text{var-stmt} \rangle) \mathbf{;} \end{aligned}$$

⟨stmt⟩ ⊨ ⟨matched-stmt⟩ | ⟨unmatched-stmt⟩
⟨matched-stmt⟩ ⊨ ⟨matched-if-else⟩ | ⟨com-stmt⟩
⟨unmatched-stmt⟩ ⊨ ⟨if-stmt⟩ | ⟨unmatched-if-else⟩

1.3.1 IO

⟨read-stmt⟩ ⊨ **read** ⟨identifier⟩
⟨write-stmt⟩ ⊨ **print** ⟨expression⟩

1.3.2 Variables

⟨var-stmt-list⟩ ⊨ ⟨var-stmt⟩{**;**⟨var-stmt⟩}**;**
⟨var-stmt⟩ ⊨ ⟨var-decl-stmt⟩ | ⟨var-attr-stmt⟩
⟨var-decl-stmt⟩ ⊨ ⟨id-list⟩**:**⟨type⟩[⟨var-attr⟩]
⟨var-attr-stmt⟩ ⊨ ⟨id-list⟩⟨var-attr⟩
⟨var-attr⟩ ⊨ **=**⟨expr-list⟩

1.4 Subprograms

⟨subprog-decl⟩ ⊨ **sub**⟨identifier⟩**(**⟨parameters⟩**)**⟨stmt-block⟩
⟨parameters⟩ ⊨ ⟨var-stmt⟩{**;**⟨var-stmt⟩}
⟨subprog-call⟩ ⊨ ⟨identifier⟩**(**⟨expr-list⟩**)**

2 Control Structures

2.1 If-else statements

⟨if-expr⟩ ⊨ **if (**⟨expression⟩**)**
⟨if-stmt⟩ ⊨ ⟨if-expr⟩ ⟨stmt⟩**;**
⟨unmatched-if-else⟩ ⊨ ⟨if-expr⟩ ⟨matched-stmt⟩**; else** ⟨unmatched-stmt⟩**;**
⟨matched-if-else⟩ ⊨ ⟨if-expr⟩ ⟨block-or-matched⟩ **else** ⟨block-or-matched⟩ | ⟨if-expr⟩ ⟨stmt-block⟩

3 Types

⟨type-list⟩ ⊨ ⟨type⟩{**,**⟨type⟩}
⟨type⟩ ⊨ ⟨native-type⟩ | ⟨user-type-id⟩
⟨native-type⟩ ⊨ ⟨primitive-type⟩ | ⟨composite-type⟩
⟨primitive-type⟩ ⊨ **int** | **float** | **char** | **string**
⟨composite-type⟩ ⊨ **[**⟨type⟩**]** | **|**⟨type⟩**|** | **(**⟨type⟩**,**⟨type-list⟩**)** | ⟨graph-type⟩
⟨graph-type⟩ ⊨ **<**⟨type⟩**>** | **<**⟨type⟩**,**⟨type⟩**>**
⟨type-decl⟩ ⊨ ⟨user-type-id⟩**{**⟨var-stmt-list⟩**}**

Observations

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

4 Expressions

⟨expression⟩ ⊨ ⟨logical-xor-expr⟩
⟨logical-xor-expr⟩ ⊨ ⟨logical-or-expr⟩ | ⟨logical-or-expr⟩⟨logical-xor-expr-aux⟩
⟨logical-xor-expr-aux⟩ ⊨ **xor** ⟨logical-or-expr⟩ | **xor** ⟨logical-or-expr⟩⟨logical-xor-expr-aux⟩
⟨logical-or-expr⟩ ⊨ ⟨logical-and-expr⟩ | ⟨logical-and-expr⟩⟨logical-or-expr-aux⟩
⟨logical-or-expr-aux⟩ ⊨ **or** ⟨logical-and-expr⟩ | **or** ⟨logical-and-expr⟩⟨logical-or-expr-aux⟩
⟨logical-and-expr⟩ ⊨ ⟨equality-expr⟩ | ⟨equality-expr⟩⟨logical-and-expr-aux⟩
⟨logical-and-expr-aux⟩ ⊨ **and** ⟨equality-expr⟩ | **and** ⟨equality-expr⟩⟨logical-and-expr-aux⟩
⟨equality-expr⟩ ⊨ ⟨rel-expr⟩ | ⟨rel-expr⟩⟨rel-expr-aux⟩
⟨equality-expr-aux⟩ ⊨ ⟨equality-op⟩ ⟨rel-expr⟩ | ⟨equality-op⟩ ⟨rel-expr⟩⟨equality-expr-aux⟩
⟨rel-expr⟩ ⊨ ⟨add-expr⟩⟨rel-expr-aux⟩
⟨rel-expr-aux⟩ ⊨ ⟨rel-op⟩⟨add-expr⟩ | ⟨rel-op⟩⟨add-expr⟩⟨rel-expr-aux⟩
⟨add-expr⟩ ⊨ ⟨mult-expr⟩ | ⟨mult-expr⟩⟨add-expr-aux⟩
⟨add-expr-aux⟩ ⊨ ⟨add-op⟩⟨mult-expr⟩ | ⟨add-op⟩⟨mult-expr⟩⟨add-expr-aux⟩
⟨mult-expr⟩ ⊨ ⟨exp-expr⟩ | ⟨exp-expr⟩⟨mult-expr-aux⟩
⟨mult-expr-aux⟩ ⊨ ⟨mult-op⟩⟨exp-expr⟩ | ⟨mult-op⟩⟨exp-expr⟩⟨mult-expr-aux⟩
⟨exp-expr⟩ ⊨ ⟨cast-expr⟩ | ⟨cast-expr⟩⟨exp-op⟩⟨exp-expr⟩
⟨cast-expr⟩ ⊨ ⟨unary-expr⟩ | ⟨unary-expr⟩⟨cast-expr-aux⟩
⟨cast-expr-aux⟩ ⊨ **@**⟨type⟩ | **@**⟨type⟩⟨cast-expr-aux⟩
⟨unary-expr⟩ ⊨ ⟨unary-op⟩⟨cast-expr⟩ | ⟨postfix-expr⟩
⟨postfix-expr⟩ ⊨ ⟨primary-expr⟩ | ⟨ident⟩**|**⟨expression⟩**|** |
⟨ident⟩**<**⟨expression⟩**>** |
⟨ident⟩**[**⟨expression⟩**]** |
⟨ident⟩**{**⟨ident⟩**}** |
⟨ident⟩**.**⟨expression⟩
⟨primary-expr⟩ ⊨ **(**⟨expression⟩**)** | ⟨ident⟩ | ⟨subprogcall⟩ | ⟨constant⟩
⟨constant⟩ ⊨ ⟨int-lit⟩ | ⟨float-lit⟩ | ⟨string-lit⟩ | ⟨bool-lit⟩ | ⟨list-lit⟩ | ⟨graph-lit⟩
⟨rel-op⟩ ⊨ > | < | <= | >=
⟨equality-op⟩ ⊨ == | **!** =
⟨unary-op⟩ ⊨ + | −
⟨add-op⟩ ⊨ + | **-**
⟨mult-op⟩ ⊨ ***** | / | % | ++ | ******
⟨exp-op⟩ ⊨ ^