**XI. Context Free Grammar**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Production** | **→** | **Production Set** |
|  | <program> | → | <loading><convardec><functiondec> state main() {<statements>} |
|  | <loading> | → | load <engine>. <loading> |
|  | <loading> | → | λ |
|  | <engine> | → | Array\_Engine |
|  | <engine> | → | Math\_Engine |
|  | <engine> | → | String\_Engine |
|  | <convardec> | → | <constdec><vartype>. <convardec> |
|  | <convardec> | → | λ |
|  | <constdec> | → | const |
|  | <constdec> | → | λ |
|  | <vardec> | → | <vartype>. <vardec> |
|  | <vardec> | → | λ |
|  | <varinit> | → | id <assignmentop> <varinittype>. <varinit> |
|  | <varinit> | → | λ |
|  | <assignmentop> | → | = |
|  | <assignmentop> | → | += |
|  | <assignmentop> | → | -= |
|  | <assignmentop> | → | /= |
|  | <assignmentop> | → | %= |
|  | <assignmentop> | → | \*= |
|  | <vartype> | → | int id<initializeINT><multipledecINT> |
|  | <vartype> | → | char id<initializeCHAR><multipledecCHAR> |
|  | <vartype> | → | float id<initializeFLOAT><multipledecFLOAT> |
|  | <vartype> | → | string id<initializeSTRING><multipledecSTRING> |
|  | <vartype> | → | boolean id<initializeBOOLEAN><multipledecBOOLEAN> |
|  | <vartype> | → | byte id<initializeBYTE><multipledecBYTE> |
|  | <varinittype> | → | <intvalue><multipleinitINT> |
|  | <varinittype> | → | <charvalue><multipleinitCHAR> |
|  | <varinittype> | → | <floatvalue><multipleinitFLOAT> |
|  | <varinittype> | → | <stringvalue><multipleinitSTRING> |
|  | <varinittype> | → | <booleanvalue><multipleinitBOOLEAN> |
|  | <varinittype> | → | <bytevalue><multipleinitBYTE> |
|  | <multipleinitINT> | → | , id <intvalue><multipleinitINT> |
|  | <multipleinitCHAR> | → | , id <charvalue><multipleinitCHAR> |
|  | <multipleinitFLOAT> | → | , id <floatvalue><multipleinitFLOAT> |
|  | <multipleinitSTRING> | → | , id <stringvalue><multipleinitSTRING> |
|  | <multipleinitBOOLEAN> | → | , id <booleanvalue><multipleinitBOOLEAN> |
|  | <multipleinitBYTE> | → | , id <bytevalue><multipleinitBYTE> |
|  | <multipleinitINT> | → | λ |
|  | <multipleinitCHAR> | → | λ |
|  | <multipleinitFLOAT> | → | λ |
|  | <multipleinitSTRING> | → | λ |
|  | <multipleinitBOOLEAN> | → | λ |
|  | <multipleinitBYTE> | → | λ |
|  | <varinittype> | → | { <arrayinittype> |
|  | <arrayinittype> | → | <valuegroupINT> } <multiplearrayinitINT> |
|  | <arrayinittype> | → | <valuegroupCHAR> } <multiplearrayinitCHAR> |
|  | <arrayinittype> | → | <valuegroupFLOAT> } <multiplearrayinitFLOAT> |
|  | <arrayinittype> | → | <valuegroupSTRING> } <multiplearrayinitSTRING> |
|  | <arrayinittype> | → | <valuegroupBOOLEAN> } <multiplearrayinitBOOL> |
|  | <arrayinittype> | → | <valuegroupBYTE> } <multiplearrayinitBYTE> |
|  | <multiplearrayinitINT> | → | , id = { <valuegroupINT> } <multiplearrayinitINT> |
|  | <multiplearrayinitCHAR> | → | , id = { <valuegroupCHAR> } <multiplearrayinitCHAR> |
|  | <multiplearrayinitFLOAT> | → | , id = { <valuegroupFLOAT> } <multiplearrayinitFLOAT> |
|  | <multiplearrayinitSTRING> | → | , id = { <valuegroupSTRING> } <multiplearrayinitSTRING> |
|  | <multiplearrayinitBOOL> | → | , id = { <valuegroupBOOL> } <multiplearrayinitBOOL> |
|  | <multiplearrayinitBYTE> | → | , id = { <valuegroupBYTE> } <multiplearrayinitBYTE> |
|  | <multipledecINT> | → | , id <initializeINT><multipledecINT> |
|  | <multipledecCHAR> | → | , id <initializeCHAR><multipledecCHAR> |
|  | <multipledecFLOAT> | → | , id<initializeFLOAT><multipledecFLOAT> |
|  | <multipledecSTRING> | → | , id<initializeSTRING><multipledecSTRING> |
|  | <multipledecBOOLEAN> | → | , id<initializeBOOLEAN><multipledecBOOLEAN> |
|  | <multipledecBYTE> | → | , id<initializeBYTE><multipledecBYTE> |
|  | <multipledecINT> | → | λ |
|  | <multipledecCHAR> | → | λ |
|  | <multipledecFLOAT> | → | λ |
|  | <multipledecSTRING> | → | λ |
|  | <multipledecBOOLEAN> | → | λ |
|  | <multipledecBYTE> | → | λ |
|  | <initializeINT> | → | <arraydecINT> |
|  | <initializeCHAR> | → | <arraydecCHAR> |
|  | <initializeFLOAT> | → | <arraydecFLOAT> |
|  | <initializeSTRING> | → | <arraydecSTRING> |
|  | <initializeBOOLEAN> | → | <arraydecBOOLEAN> |
|  | <initializeBYTE> | → | <arraydecBYTE> |
|  | <initializeINT> | → | = <intvalue> |
|  | <initializeCHAR> | → | = <charvalue> |
|  | <initializeFLOAT> | → | = <floatvalue> |
|  | <initializeSTRING> | → | = <stringvalue> |
|  | <initializeBOOLEAN> | → | = <booleanvalue> |
|  | <initializeBYTE> | → | = <bytevalue> |
|  | <initializeINT> | → | λ |
|  | <initializeCHAR> | → | λ |
|  | <initializeFLOAT> | → | λ |
|  | <initializeSTRING> | → | λ |
|  | <initializeBOOLEAN> | → | λ |
|  | <initializeBYTE> | → | λ |
|  | <arraydecINT> | → | [<arraysize>]<index><index><index> <arrayvalueINT> |
|  | <arraydecCHAR> | → | [<arraysize>]<index><index><index> <arrayvalueCHAR> |
|  | <arraydecFLOAT> | → | [<arraysize>]<index><index><index> <arrayvalueFLOAT> |
|  | <arraydecSTRING> | → | [<arraysize>]<index><index><index> <arrayvalueSTRING> |
|  | <arraydecBOOLEAN> | → | [<arraysize>]<index><index><index> <arrayvalueBOOLEAN> |
|  | <arraydecBYTE> | → | [<arraysize>]<index><index><index> <arrayvalueBYTE> |
|  | <index> | → | [<arraysize>] |
|  | <index> | → | λ |
|  | <arraysize> | → | <bytevalue> |
|  | <arraysize> | → | <intvalue> |
|  | <crementid> | → | <crementop>id |
|  | <crementid> | → | id<crementop> |
|  | <crementop> | → | ++ |
|  | <crementop> | → | -- |
|  | <arrayvalueINT> | → | = { <valuegroupINT> } |
|  | <arrayvalueCHAR> | → | = { <valuegroupCHAR> } |
|  | <arrayvalueFLOAT> | → | = { <valuegroupFLOAT> } |
|  | <arrayvalueSTRING> | → | = { <valuegroupSTRING> } |
|  | <arrayvalueBOOLEAN> | → | = { <valuegroupBOOLEAN> } |
|  | <arrayvalueBYTE> | → | = { <valuegroupBYTE> } |
|  | <arrayvalueINT> | → | λ |
|  | <arrayvalueCHAR> | → | λ |
|  | <arrayvalueFLOAT> | → | λ |
|  | <arrayvalueSTRING> | → | λ |
|  | <arrayvalueBOOLEAN> | → | λ |
|  | <arrayvalueBYTE> | → | λ |
|  | <valuegroupINT> | → | ( <valuegroupINT> ) <valuegroupINT> |
|  | <valuegroupCHAR> | → | ( <valuegroupCHAR> ) <valuegroupCHAR> |
|  | <valuegroupFLOAT> | → | ( <valuegroupFLOAT> ) <valuegroupFLOAT> |
|  | <valuegroupSTRING> | → | ( <valuegroupSTRING> ) <valuegroupSTRING> |
|  | <valuegroupBOOLEAN> | → | ( <valuegroupBOOLEAN> ) <valuegroupBOOLEAN> |
|  | <valuegroupBYTE> | → | ( <valuegroupBYTE> ) <valuegroupBYTE> |
|  | <valuegroupINT> | → | <intvalue> <int\_tail> |
|  | <valuegroupCHAR> | → | <charvalue> <char\_tail> |
|  | <valuegroupFLOAT> | → | <floatvalue> <float\_tail> |
|  | <valuegroupSTRING> | → | <stringvalue> <string\_tail> |
|  | <valuegroupBOOLEAN> | → | <booleanvalue> <boolean\_tail> |
|  | <valuegroupBYTE> | → | <bytevalue> <byte\_tail> |
|  | <intvalue> | → | intlit |
|  | <intvalue> | → | <expressionid> |
|  | <intvalue> | → | <mathopINT> |
|  | <intvalue> | → | ~intlit |
|  | <bytevalue> | → | bytelit |
|  | <bytevalue> | → | <expressionid> |
|  | <bytevalue> | → | <mathopBYTE> |
|  | <bytevalue> | → | ~bytelit |
|  | <charvalue> | → | charlit |
|  | <charvalue> | → | id |
|  | <floatvalue> | → | floatlit |
|  | <floatvalue> | → | <expressionid> |
|  | <floatvalue> | → | <mathopFLOAT> |
|  | <floatvalue> | → | ~id |
|  | <stringvalue> | → | stringlit |
|  | <stringvalue> | → | id |
|  | <booleanvalue> | → | booleanlit |
|  | <booleanvalue> | → | id |
|  | <booleanvalue> | → | <expressionBOOL> |
|  | <booleanvalue> | → | ~booleanlit |
|  | <expressionid> | → | <crementid> |
|  | <expressionid> | → | id |
|  | <int\_tail> | → | , <intvalue> <int\_tail> |
|  | <char\_tail> | → | , <charvalue> <char\_tail> |
|  | <float\_tail> | → | , <floatvalue> <float\_tail> |
|  | <string\_tail> | → | , <stringvalue> <string\_tail> |
|  | <boolean\_tail> | → | , <booleanvalue> <boolean\_tail> |
|  | <byte\_tail> | → | , <bytevalue> <byte\_tail> |
|  | <int\_tail> | → | λ |
|  | <char\_tail> | → | λ |
|  | <float\_tail> | → | λ |
|  | <string\_tail> | → | λ |
|  | <boolean\_tail> | → | λ |
|  | <byte\_tail> | → | λ |
|  | <mathopINT> | → | <intvalue><mathopINT\_tail> |
|  | <mathopINT> | → | ( <mathopINT> ) <mathopINT\_tail> |
|  | <mathopINT\_tail> | → | <operators><mathopINT> |
|  | <mathopINT\_tail> | → | λ |
|  | <mathopBYTE> | → | <bytevalue><mathopBYTE\_tail> |
|  | <mathopBYTE> | → | ( <mathopBYTE> ) <mathopBYTE\_tail> |
|  | <mathopBYTE\_tail> | → | <operators><mathopBYTE> |
|  | <mathopBYTE\_tail> | → | λ |
|  | <mathopFLOAT> | → | <floatvalue><mathopFLOAT\_tail> |
|  | <mathopFLOAT> | → | ( <mathopFLOAT> ) <mathopFLOAT\_tail> |
|  | <mathopFLOAT\_tail> | → | <operators><mathopFLOAT> |
|  | <mathopFLOAT\_tail> | → | λ |
|  | <operators> | → | <arithmeticop> |
|  | <operators> | → | <bitwiseop> |
|  | <arithmeticop> | → | + |
|  | <arithmeticop> | → | - |
|  | <arithmeticop> | → | / |
|  | <arithmeticop> | → | \* |
|  | <arithmeticop> | → | % |
|  | <bitwiseop> | → | & |
|  | <bitwiseop> | → | | |
|  | <bitwiseop> | → | ^ |
|  | <expressionBOOL> | → | <logicexpression><expressionBOOL\_tail> |
|  | <expressionBOOL> | → | (<logicexpression>) <expressionBOOL\_tail> |
|  | <expressionBOOL\_tail> | → | <logicalop><expressionBOOL> |
|  | <logicexpression> | → | booleanlit |
|  | <logicexpression> | → | <relexpression> |
|  | <logicexpression> | → | !<logicexpression> |
|  | <logicalop> | → | && |
|  | <logicalop> | → | || |
|  | <relexpression> | → | <mathop><relexpression\_tail> |
|  | <relexpression> | → | (<mathop>) <relexpression\_tail> |
|  | <relexpression\_tail> | → | <relationop><relexpression> |
|  | <relationop> | → | > |
|  | <relationop> | → | < |
|  | <relationop> | → | >= |
|  | <relationop> | → | <= |
|  | <relationop> | → | == |
|  | <relationop> | → | != |
|  | <mathop> | → | <mathopINT> |
|  | <mathop> | → | <mathopBYTE> |
|  | <mathop> | → | <mathopFLOAT> |
|  | <mathop> | → | <expressionid> |
|  | <functiondec> | → | <functionVOID> |
|  | <functiondec> | → | <functionINT> |
|  | <functiondec> | → | <functionCHAR> |
|  | <functiondec> | → | <functionBYTE> |
|  | <functiondec> | → | <functionSTRING> |
|  | <functiondec> | → | <functionFLOAT> |
|  | <functiondec> | → | <functionBOOLEAN> |
|  | <functionVOID> | → | void map id { <fbodyVOID> } |
|  | <functionINT> | → | int map id { <fbodyINT> } |
|  | <functionCHAR> | → | char map id { <fbodyCHAR> } |
|  | <functionBYTE> | → | byte map id { <fbodyBYTE> } |
|  | <functionSTRING> | → | string map id { <fbodySTRING> } |
|  | <functionFLOAT> | → | float map id { <fbodyFLOAT> } |
|  | <functionBOOLEAN> | → | boolean map id { <fbodyBOOLEAN> } |
|  | <fbodyVOID> | → | <statements><fbodyVOID> |
|  | <fbodyVOID> | → | return. <fbodyVOID> |
|  | <fbodyVOID> | → | λ |
|  | <fbodyINT> | → | <statements><fbodyINT> |
|  | <fbodyINT> | → | return <mathopINT>. <fbodyINT> |
|  | <fbodyINT> | → | λ |
|  | <fbodyCHAR> | → | <statements><fbodyCHAR> |
|  | <fbodyCHAR> | → | return <charvalue>. <fbodyCHAR> |
|  | <fbodyCHAR> | → | λ |
|  | <fbodySTRING> | → | <statements><fbodySTRING> |
|  | <fbodySTRING> | → | return <stringvalue>. <fbodySTRING> |
|  | <fbodySTRING> | → | λ |
|  | <fbodyBYTE> | → | <statements><fbodyBYTE> |
|  | <fbodyBYTE> | → | return <mathopBYTE>. <fbodyBYTE> |
|  | <fbodyBYTE> | → | λ |
|  | <fbodyFLOAT> | → | <statements><fbodyFLOAT> |
|  | <fbodyFLOAT> | → | return <mathopFLOAT>. <fbodyFLOAT> |
|  | <fbodyFLOAT> | → | λ |
|  | <fbodyBOOLEAN> | → | <statements><fbodyBOOLEAN> |
|  | <fbodyBOOLEAN> | → | return <expressionBOOL>. <fbodyBOOLEAN> |
|  | <fbodyBOOLEAN> | → | λ |
|  | <statements> | → | <crementid>. <statements> |
|  | <statements> | → | <vardec> <statements> |
|  | <statements> | → | <varinit> <statements> |
|  | <statements> | → | <outputstat> <statements> |
|  | <statements> | → | <ifelse> <statements> |
|  | <statements> | → | <inputstat> <statements> |
|  | <statements> | → | <switch> <statements> |
|  | <statements> | → | <do> <statements> |
|  | <statements> | → | <while> <statements> |
|  | <statements> | → | <for> <statements> |
|  | <statements> | → | λ |
|  | <outputstat> | → | cast(<output>). |
|  | <output> | → | <mathop><concat> |
|  | <output> | → | <charvalue><concat> |
|  | <output> | → | <stringvalue><concat> |
|  | <concat> | → | + <output> <concat> |
|  | <concat> | → | λ |
|  | <inputstat> | → | id = absorb(). |
|  | <ifelse> | → | if (<expressionBOOL>) {<statements>} <elseif><else> |
|  | <elseif> | → | else if (<expressionBOOL>) {<statements>} <elseif> |
|  | <elseif> | → | λ |
|  | <else> | → | else {<statements>} |
|  | <else> | → | λ |
|  | <switch> | → | switch(id) { <casechoices><defaultcase> } |
|  | <casechoices> | → | case <casetype> |
|  | <casetype> | → | <caseint>: <statements>. break. <acondistINT> |
|  | <casetype> | → | <casechar>: <statements>. break. <acondistCHAR> |
|  | <casetype> | → | <caseboolean>: <statements>. break. <acondistBOOLEAN> |
|  | <casetype> | → | <casestring>: <statements>. break. <acondistSTRING> |
|  | <casetype> | → | <casebyte>: <statements>. break. <acondistBYTE> |
|  | <casetype> | → | <casefloat>: <statements>. break. <acondistFLOAT> |
|  | <acondistINT> | → | case <caseint>: <statements>. break. <acondistINT> |
|  | <acondistINT> | → | λ |
|  | <acondistCHAR> | → | case <casechar>: <statements>. break. <acondistCHAR> |
|  | <acondistCHAR> | → | λ |
|  | <acondistBOOLEAN> | → | case <caseboolean>: <statements>. break. <acondistBOOLEAN> |
|  | <acondistBOOLEAN> | → | λ |
|  | <acondistSTRING> | → | case <casestring>: <statements>. break. <acondistSTRING> |
|  | <acondistSTRING> | → | λ |
|  | <acondistBYTE> | → | case <casebyte>: <statements>. break. <acondistBYTE> |
|  | <acondistBYTE> | → | λ |
|  | <acondistFLOAT> | → | case <casefloat>: <statements>. break. <acondistFLOAT> |
|  | <acondistFLOAT> | → | λ |
|  | < defaultcase > | → | default: <statements>. break. |
|  | < defaultcase > | → | λ |
|  | <caseint> | → | intlit |
|  | <caseint> | → | id |
|  | <casefloat> | → | charlit |
|  | <casefloat> | → | id |
|  | <casechar> | → | charlit |
|  | <casechar> | → | id |
|  | <casebyte> | → | bytelit |
|  | <casebyte> | → | id |
|  | <casestring> | → | stringlit |
|  | <casestring> | → | id |
|  | <caseboolean> | → | booleanlit |
|  | <caseboolean> | → | id |
|  | <for> | → | for (<initializeNUM>. <expressionBOOL>. <crementid>) {<statements>} |
|  | <initializeNUM> | → | <varinit> |
|  | <initializeNUM> | → | int id = <mathopINT><mupltipleinitializeINT> |
|  | <initializeNUM> | → | byte id = <mathopBYTE><mupltipleinitializeBYTE> |
|  | <initializeNUM> | → | float id = <mathopFLOAT><mupltipleinitializeFLOAT> |
|  | <mupltipleinitializeINT> | → | , id = <mathopINT><mupltipleinitializeINT> |
|  | <mupltipleinitializeBYTE> | → | , id = <mathopBYTE><mupltipleinitializeBYTE> |
|  | <mupltipleinitializeFLOAT> | → | , id = <mathopFLOAT><mupltipleinitializeFLOAT> |
|  | <while> | → | while (<expressionBOOL>) {<statements>} |
|  | <do> | → | do {<statements>} while (<expressionBOOL>). |