Delimiters

⟨*singleQuote*⟩ ::= '

⟨*doubleQuote*⟩ ::= "

⟨*terminator*⟩ ::= ‘`;`’

Primitive Types

⟨*int*⟩ ::= [integer]

⟨*bool*⟩ ::= ‘`true`’ | ‘`false`’

⟨*char*⟩ ::= ⟨*singleQuote*⟩[character]⟨*singleQuote*⟩

⟨*string*⟩ ::= ⟨*doubleQuote*⟩[character*]⟨*doubleQuote*⟩

⟨*null*⟩ ::= ‘`null`’

Types and Typeclasses

⟨*alias*⟩ ::= ‘`alias`’ ⟨*ident*⟩ ‘`=`’ ⟨*type*⟩

⟨*lowerBoundOp*⟩ ::= ‘`:>`’

⟨*upperBoundOp*⟩ ::= ‘`<:`’

⟨*bounding*⟩ ::= [⟨*lowerBoundOp*⟩⟨*ident*⟩][⟨*upperBoundOp*⟩⟨*ident*⟩]

⟨*generics*⟩ ::= ‘`[`’⟨*ident*⟩⟨*bounding*⟩[‘`,`’ ⟨*ident*⟩⟨*bounding*⟩]*‘`]`’

⟨*genericIdent*⟩ ::= ⟨*ident*⟩[⟨*generics*⟩]

⟨*constructor*⟩ ::= ⟨*ident*⟩[‘`,`’⟨*ident*⟩]*

⟨*cosntructorList*⟩ ::= ⟨*constructor*⟩[‘`|`’⟨*constructor*⟩]

⟨*adt*⟩ ::= ‘`type`’ ⟨*genericIdent*⟩‘{’[⟨*cosntructorLists*⟩]‘}’⟨*terminator*⟩⟨*exp*⟩

⟨*members*⟩ ::= ⟨*ident*⟩‘`:`’⟨*type*⟩[‘`,`’⟨*ident*⟩‘`:`’⟨*type*⟩]*

⟨*record*⟩ ::= ‘`record`’ ⟨*genericIdent*⟩‘{’[⟨*members*⟩]‘}’⟨*terminator*⟩⟨*exp*⟩

⟨*signatures*⟩ ::= ⟨*ident*⟩‘`=`’⟨*type*⟩[‘`,`’⟨*ident*⟩‘`=`’⟨*type*⟩]*

⟨*typeclass*⟩ ::= ‘`typeclass`’ ⟨*genericIdent*⟩[‘`=>`’ ⟨*ident*⟩]‘{’[⟨*signatures*⟩]‘}’⟨*terminator*⟩⟨*exp*⟩

⟨*instance*⟩ ::= ‘`instance`’ ⟨*ident*⟩‘`:`’⟨*ident*⟩‘{’[⟨*funDef*⟩]*‘}’⟨*terminator*⟩⟨*exp*⟩

Type Definitions

⟨*type*⟩ ::= ‘`int`’ | ‘`bool`’ | ‘`char`’ | ‘`string`’ | ‘`null`’
| ⟨*type*⟩ ‘`->`’ ⟨*type*⟩
| ‘`(`’[⟨*type*⟩[‘`,`’ ⟨*type*⟩]*]‘`)`’ ‘`->`’ ‘`(`’[⟨*type*⟩[‘`,`’ ⟨*type*⟩]*]‘`)`’
| ‘`List`’ | ‘`Array`’ | ‘`Set`’ ‘`[`’⟨*type*⟩‘`]`’
| ‘`Tuple`’ ‘`[`’⟨*type*⟩[‘`,`’ ⟨*type*⟩]*‘`]`’
| ‘`Dict`’ ‘`[`’⟨*type*⟩‘`,`’ ⟨*type*⟩‘`]`’
| ⟨*ident*⟩[‘`[`’⟨*type*⟩‘`]`’]

Arithmetic and Boolean Operators

⟨*arithOp*⟩ ::= ‘`+`’ | ‘`-`’ | ‘`*`’ | ‘`/`’ | ‘`%`’

⟨*boolOp*⟩ ::= ‘`<`’ | ‘`>`’ | ‘`<=`’ | ‘`>=`’ | ‘`!`’ | ‘`!=`’ | ‘`==`’ | ‘`&&`’ | ‘`||`’

⟨*op*⟩ ::= ⟨*arithOp*⟩ | ⟨*boolOp*⟩

Functions

⟨*param*⟩ ::= ⟨*ident*⟩‘`:`’⟨*type*⟩[‘`=`’ ⟨*atom*⟩ | ⟨*collection*⟩]

⟨*funDef*⟩ ::= ‘`fn`’ ⟨*genericIdent*⟩‘`(`’[⟨*param*⟩ [‘`,`’ ⟨*param*⟩]*]‘`)`’[‘`->`’⟨*type*⟩]‘`=`’⟨*smp*⟩⟨*terminator*⟩

⟨*prog*⟩ ::= [⟨*funDef*⟩]*⟨*exp*⟩

⟨*app*⟩ ::= ⟨*atom*⟩[[‘`[`’⟨*type*⟩[‘`,`’ ⟨*type*⟩]*‘`]`’]* [‘`(`’[⟨*smp*⟩[‘`,`’ ⟨*smp*⟩]*]‘`)`’]*]

⟨*lambda*⟩ ::= ‘`|`’[⟨*param*⟩ [‘`,`’ ⟨*param*⟩]*]‘`|`’[‘`->`’⟨*type*⟩]‘`=`’⟨*smp*⟩⟨*terminator*⟩

Pattern Matching and Switches

⟨*value*⟩ ::= ⟨*ident*⟩‘`(`’[⟨*value*⟩[‘`,`’⟨*value*⟩]]‘`)`’
| ⟨*prim*⟩
| ‘_’

⟨*caseVal*⟩ ::= ⟨*ident*⟩‘`:`’⟨*type*⟩
| ⟨*value*⟩

⟨*match*⟩ ::= ‘`match`’‘`(`’⟨*smp*⟩‘`)`’ ‘`{`’‘`case`’⟨*caseVal*⟩‘`=>`’⟨*smp*⟩[‘`case`’⟨*caseVal*⟩‘`=>`’⟨*smp*⟩]*‘`}`’

⟨*matchSwitch*⟩ ::= ⟨*match*⟩ | ⟨*switch*⟩

Expressions

⟨*atom*⟩ ::= ⟨*int*⟩ | ⟨*bool*⟩ | ⟨*char*⟩ | ⟨*string*⟩ | ⟨*null*⟩
  | '`(`'⟨*smp*⟩'`)`'
  | ⟨*ident*⟩['`.`'⟨*ident*⟩]*

⟨*tight*⟩ ::= ⟨*app*⟩['`|>`'⟨*app*⟩]
  | '{'⟨*exp*⟩'}'

⟨*utight*⟩ ::= [⟨*op*⟩]⟨*tight*⟩

⟨*smp*⟩ ::= ⟨*utight*⟩[⟨*op*⟩⟨*utight*⟩]
  | '`if`' '`(`'⟨*smp*⟩'`)`' ⟨*smp*⟩ ['`else`' ⟨*smp*⟩]
  | '`List`' | '`Tuple`' | '`Array`' | '`Set`' '{'[⟨*smp*⟩['`,`' ⟨*smp*⟩]*]'}'
  | '`Dict`' '{'[⟨*smp*⟩'`:`'⟨*smp*⟩['`,`' ⟨*smp*⟩'`:`'⟨*smp*⟩]*]'}'
  | ⟨*matchSwitch*⟩
  | ⟨*typeclass*⟩
  | ⟨*instance*⟩
  | ⟨*alias*⟩
  | ⟨*adt*⟩
  | ⟨*record*⟩
  | ⟨*prog*⟩
  | ⟨*lambda*⟩

⟨*exp*⟩ ::= ⟨*smp*⟩[⟨*terminator*⟩⟨*exp*⟩]
  | ['`lazy`'] '`val`' ⟨*ident*⟩['`:`'⟨*type*⟩] '`=`' ⟨*smp*⟩⟨*terminator*⟩⟨*exp*⟩
  | '`include`' ⟨*file*⟩⟨*terminator*⟩⟨*exp*⟩