Delimiters

⟨*singleQuote*⟩ ::= '

⟨*doubleQuote*⟩ ::= "

⟨*terminator*⟩ ::= ‘`;`’ | ‘`\n`’

Primitive Types

⟨*int*⟩ ::= [integer]

⟨*bool*⟩ ::= ‘`true`’ | ‘`false`’

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

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

⟨*null*⟩ ::= ‘`null`’

Algebraic Data Types and Type-Traits

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

⟨*adt*⟩ ::= ‘`type`’ ⟨*ident*⟩[⟨*generic*⟩][‘`:`’ ⟨*ident*⟩[⟨*generic*⟩]]‘`{`’[⟨*ident*⟩‘`:`’⟨*type*⟩[‘`,`’ ⟨*ident*⟩‘`:`’⟨*type*⟩]*]‘`}`’

⟨*typeclass*⟩ ::= ‘`typeclass`’ ⟨*ident*⟩ ‘`{`’[⟨*ident*⟩‘`=`’⟨*type*⟩[‘`,`’⟨*ident*⟩‘`=`’⟨*type*⟩]*]‘`}`’

⟨*instance*⟩ ::= ‘`instance`’ ⟨*ident*⟩‘`:`’⟨*ident*⟩‘`{`’⟨*prog*⟩‘`}`’

Types

⟨*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*⟩]

⟨*lowerBoundOp*⟩ ::= ‘`:>`’

⟨*upperBoundOp*⟩ ::= ‘`<:`’

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

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

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

⟨*prog*⟩ ::= [⟨*funDef*⟩]*

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

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

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*⟩
| ⟨*adt*⟩
| ⟨*prog*⟩
| ⟨*anonLmbd*⟩

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