Archetype by Compilers Group 2: A summary of the Lexical and Syntactic Analysers

Abhay Shankar K: cs21btech11001 Kartheek Tammana: cs21btech11028
Prasham Walvekar: cs21btech11047 Sumedh Kashikar: es21btech11033

Keywords and their correspnding tokens

Where unspecified, the keyword is used in a manner identical to C.

Name	Token	Use
`claim`	KW_CLAIM	To claim a variable is an Archetype
`is`	KW_IS	To claim a variable is an Archetype
`Group`	KW_GROUP	Archetype name, also used in morphisms
`Ring`	KW_RING	Archetype name, also used in morphisms
`Field`	KW_FIELD	Archetype name
`Space`	KW_SPACE	Archetype name
`print`	KW_PRINT	To print a variable
`let`	KW_LET	To declare a variable
`return`	KW_RETURN	To return a value from a function
`if`	KW_IF	
`else`	KW_ELSE	
`while`	KW_WHILE	
`for`	KW_FOR	
`in`	KW_IN	To iterate over a range
`switch`	KW_SWITCH	
`case`	KW_CASE	
`default`	KW_DEFAULT	
`break`	KW_BREAK	
`continue`	KW_CONTINUE	
`fn`	KW_FN	To declare a function
`morph`	KW_MORPH	To declare a morphism
`forge`	KW_FORGE	To declare a forge
`struct`	KW_STRUCT	
`enum`	KW_ENUM	
`as`	KW_AS	Forging
`true`	KW_TRUE	
`false`	KW_FALSE	

Data types and their corresponding tokens

Name	Token
`Cyclic<>`	KW_CYCLIC
`BigRational`	KW_BIG_RATIONAL
`Complex`	KW_COMPLEX
`Symmetric<>`	KW_SYMMETRIC

Name	Token
`Alternating<>`	KW_ALTERNATING
`Dihedral<>`	KW_DIHEDRAL
`InvMat<>`	KW_INV_MAT
`BigInt`	KW_BIGINT
`Matrix<>`	KW_MATRIX
`Polynomial<>`	KW_POLYNOMIAL
`Vec<>`	KW_VEC
`Buf<>`	KW_BUF
(Un)Signed integers, char, float, bool and str	PRIMITIVE_DTYPE

Operators and their corresponding tokens

Single character operators do not have a correspoonding token, and are represented sing their values.

Value	Use
`+`	Addition if type claims group
`-`	Subtraction if type claims group, unary negation
`*`	Multiplication if type claims ring, dereferencing
`/`	Division if type claims field
`%`	Modulus for `System` integer types
`@`	Dot product
`.`	Field access
`[]`	Indexing - not technically an operator
`()`	Function call - not technically an operator
`=`	Assignment
`!`	Logical negation
`>`, `<`	

Value	Token	Use
`=>`	ARROW	Claim-internal mapping
`::`	VARIANT	Enum variants
`..`	SLICE	Similar to python
`++`	INCR	
`--`	DECR	
`==`	rel_op	
`&&`	rel_op	
`||`	rel_op	