Function call, nested variable assignment, and inc\_dec

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <var\_fn\_assign> | {ID}  {self}  {grand}  {ID, self, grand} |  |
| <var\_fn\_assign1> | {[}  {., inc\_dec, =, compound\_assignment}  {(}  {[, ., inc\_dec, =, compound\_assignment, (} | {;} |
| <var\_fn\_assign2> | {.}  {inc\_dec}  {=, compound\_assignment}  {., inc\_dec, =, compound\_assignment} | {;} |
| <var\_fn\_assign3> | {.}  {[}  {ϵ}  {., [, ϵ} | {;} |
| <var\_fn\_assign4> | {ID} | {;} |
| <this\_super> | {self}  {grand}  {self, grand} |  |
| <this\_super1> | {(}  {.}  {(, .} |  |

Assignment right side,

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <assign1> | {ID}  {self, grand}  {(, !, int\_const, float\_const, char\_const, str\_const, bool\_const, PM, make}  {inc\_dec}  {ID, self, grand, (, !, int\_const, float\_const, char\_const, str\_const, bool\_const, PM, make, inc\_dec} | {;} |
| <exp1> | {(}  {!}  {int\_const, float\_const, char\_const, str\_const, bool\_const}  {PM}  {make}  {(, !, int\_const, float\_const, char\_const, str\_const, bool\_const, PM, make} | {;} |
| <assign2> | {[}  {., ^, MDM, PM, relational, &&, ||, inc\_dec, as, =, compound\_assignment, ϵ}  {(}  {[, ., ^, MDM, PM, relational, &&, ||, inc\_dec, as, =, compound\_assignment, (, ϵ} | {;} |
| <assign3> | {.}  {^, MDM, PM, relational, &&, ||, ϵ}  {inc\_dec}  {as}  {=, compound\_assignment}  {., ^, MDM, PM, relational, &&, ||, inc\_dec, as, =, compound\_assignment, ϵ} | {;} |
| <assign4> | {.}  {^, MDM, PM, relational, &&, ||, ϵ}  {as}  {[}  {., ^, MDM, PM, relational, &&, ||, as, [, ϵ} | {;} |
| <assign5> | {ID} | {;} |

Declaration

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <dec> | {ID} | {;} |
| <dec1> | {:} | {;} |
| <dec2> | {=}  {ϵ}  {=, ϵ} | {;} |
| <dec3> | {ID, self, grand, (, !, int\_const, float\_const, char\_const, str\_const, bool\_const, PM, make, inc\_dec}  {[}  {ID, self, grand, (, !, int\_const, float\_const, char\_const, str\_const, bool\_const, PM, make, inc\_dec, [} |  |

For loop

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <for\_st> | {for} |  |
| <decs> | {ID}  {[, :}  {ID, [} | {in} |
| <des\_dec\_ref> | {[}  {:}  {[, :} | {in} |
| <des\_dec\_ref1> | {,}  {ϵ}  {,, ϵ} | {]} |
| <iterator> | {ID, self, grand, (, !, int\_const, float\_const, char\_const, str\_const, bool\_const, PM, make} | {)} |
| <range> | {range} | {)} |
| <range1> | {range}  {ϵ}  {range, ϵ} | {)} |

De-structured assignment and declaration

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <des\_dec\_assign> | {[} |  |
| <des\_this\_super> | {ID}  {this, super}  {ID, this, super} |  |
| <des\_this\_super1> | {ID}  {this, super}  {ID, this, super} |  |
| <var\_id> | {[, (, .}  {,, ]}  {[, (, ., ,,]} |  |
| <des\_dec\_assign1> | {,}  {]}  {,, ]} |  |
| <des\_dec\_assign2> | {,}  {]}  {,, ]} |  |
| <des\_dec\_assign3> | {=}  {:}  {=, :} |  |
| <des\_dec\_assign4> | {;}  {[, ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM}  {;, [, ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM} |  |

Expression

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <OE> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM} |  |
| <AE> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM} |  |
| <RE> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM} |  |
| <E> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM} |  |
| <T> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM} |  |
| <P> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM} |  |
| <F> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec}  {(}  {!}  {PM}  {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM} | {^, MDM, PM, relational, &&, ||, :, ;, ,, ->, ], ), range} |
| <B> | {(} |  |
| <OE’> | {||}  {ϵ}  {||, ϵ} | {:, ;, ,, ->, ], ), range} |
| <AE’> | {&&}  {ϵ}  {&&, ϵ} | {||, :, ;, ,, ->, ], ), range} |
| <RE’> | {relational}  {ϵ}  {relational, ϵ} | {&&, ||, :, ;, ,, ->, ], ), range} |
| <E’> | {PM}  {ϵ}  {PM, ϵ} | {relational, &&, ||, :, ;, ,, ->, ], ), range} |
| <T’> | {MDM}  {ϵ}  {MDM, ϵ} | {PM, relational, &&, ||, :, ;, ,, ->, ], ), range} |
| <P’> | {^}  {ϵ}  {^, ϵ} | {MDM, PM, relational, &&, ||, :, ;, ,, ->, ], ), range} |
| <operand> | {ID}  {self, grand}  {int\_const, float\_const, char\_const, str\_const, bool\_const}  {inc\_dec}  {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec} | {^, MDM, PM, relational, &&, ||, :, ;, ,, ->, ], ), range} |
| <unpacked> | {^, MDM, PM, relational, &&, ||, ϵ} | {;} |
| <exp> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM}  {make}  {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, make} |  |
| <exp\_array> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, make}  {[}  {[, ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, make} |  |

Operand

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <operand1> | {[}  {., inc\_dec, as, ϵ}  {(}  {[, ., inc\_dec, as, (, ϵ} | {^, MDM, PM, relational, &&, ||, :, ;, ,, ->, ], ), range} |
| <operand2> | {.}  {inc\_dec}  {as}  {ϵ}  {., inc\_dec, as, ϵ} | {^, MDM, PM, relational, &&, ||, :, ;, ,, ->, ], ), range} |
| <operand3> | {.}  {[}  {ϵ}  {., [, ϵ} | {^, MDM, PM, relational, &&, ||, :, ;, ,, ->, ], ), range} |
| <operand4> | {ID} |  |

Variable

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <var> | {ID} |  |
| <var1> | {[}  {., ϵ}  {(}  {[, ., (,ϵ} | {^, MDM, PM, relational, &&, ||, :, ;, ,, ->, ], ), range} |
| <var2> | {.}  {ϵ}  {., ϵ} | {^, MDM, PM, relational, &&, ||, :, ;, ,, ->, ], ), range} |
| <var3> | {.}  {[}  {., [} |  |
| <var4> | {[}  {.}  {(}  {[, ., (} |  |

Object declaration

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <obj\_dec> | {make} |  |
| <obj\_dec1> | {str}  {ID}  {primitive\_type}  {str, ID, primitive\_type} |  |
| <obj\_dec2> | {(}  {[}  {(, [} |  |
| <array\_ref> | {[} |  |
| <array\_ref1> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, make}  {]}  {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, ], make} |  |
| <array\_ref2> | {[}  {ϵ}  {[, ϵ} | {:, ;, ,, ->, ], ), range} |
| <array\_ref\_exp> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, make}  {]}  {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, ], make} |  |
| <array\_ref3> | {[}  {:}  {[, :} |  |
| <array\_ref4> | {[}  {ϵ}  {[, ϵ} | {:, ;, ,, ->, ], ), range} |

Array index

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <array\_index> | {[} |  |
| <array\_index1> | {[}  {ϵ}  {[, ϵ} | {^, MDM, PM, relational, &&, ||, :, ;, ,, ->, ], ), range, ., inc\_dec, =, compound\_assignment, as} |

Arguments

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <args> | {(} |  |
| <args1> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, make}  {)}  {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, ), make} |  |
| <args2> | {,}  {)}  {,, )} |  |

Data type

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <type> | {str}  {ID}  {primitive\_type}  {str, ID, primitive\_type} |  |
| <dim> | {[}  {ϵ}  {[, ϵ} | {^, MDM, PM, relational, &&, ||, =, in, range, ,, ), {, } |

Single line and multiline statement

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <SST> | {ID}  {const}  {self, grand}  {inc\_dec}  {if}  {match}  {while}  {for}  {try}  {return}  {continue\_break}  {[}  {ID, const, self, grand, inc\_dec, if, match, while, for, try, return, continue\_break, [} |  |
| <SST1> | {:}  {[, ., inc\_dec, =, (}  {:, [, ., inc\_dec, =, (} |  |
| <SST2> | {ID}  {[}  {ID, [} |  |
| <MST> | {ID, const, self, grand, inc\_dec, if, match, while, for, try, return, continue\_break, [}  {ϵ}  {ID, const, self, grand, inc\_dec, if, match, while, for, try, return, continue\_break, [, ϵ} | {}} |

Body

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <body> | {;}  {ID, const, self, grand, inc\_dec, if, match, while, for, try, return, continue\_break, [}  {{} |  |

Array declaration

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <array\_dec> | {[} |  |
| <array\_dec1> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, make}  {]}  {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, ], make} |  |
| <array\_dec2> | {,}  {]}  {,, ]} |  |

Match case

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <match\_st> | {match} |  |
| <match\_body> | {case}  {default}  {{}  {case, default, {} |  |
| <case> | {case} |  |
| <default> | {default} |  |
| <case\_default> | {case}  {default}  {ϵ}  {case, default, ϵ} | {}} |

Try except

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <try\_st> | {try} |  |
| <except> | {except} |  |
| <except\_multi> | {except} |  |
| <except\_multi1> | {except}  {ϵ}  {except, ϵ} | {finally, ID, self, grand, inc\_dec, if, match, while, for, try, return, continue\_break, [, }, else} |
| <finally> | {finally}  {ϵ}  {finally, ϵ} | {ID, self, grand, inc\_dec, if, match, while, for, try, return, continue\_break, [, }, else} |

If else

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <if\_st> | {if} |  |
| <else> | {else}  {ϵ}  {else, ϵ} | {ID, self, grand, inc\_dec, if, match, while, for, try, return, continue\_break, [, }, else} |

While loop

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <while\_st> | {while} |  |

Function declaration

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <func\_dec> | {func} |  |
| <func\_dec1> | {ID} |  |
| <params> | {ID}  {)}  {ID, )} |  |
| <params1> | {,}  {)}  {,, )} |  |
| <type\_void> | {str, primitive\_type, ID}  {void}  {str, primitive\_type, ID, void} |  |

Return

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <return\_st> | {return} |  |
| <return\_exp> | {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM}  {;}  {ID, self, grand, int\_const, float\_const, char\_const, str\_const, bool\_const, inc\_dec, (, !, PM, ;} |  |

Abstract Class

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <abstract\_class\_def> | {abstract} |  |
| <abstract\_body> | {abstract}  {access\_modifier, passive, const, ID, func, [}  {ϵ}  {abstract, access\_modifier, passive, const, Id, [, func, ϵ} | {}} |
| <adecs> | {ID}  {func}  {[}  {ID, func, [} |  |
| <abstract\_body1> | {abstract}  {passive, const, ϵ}  {abstract, passive, const, ϵ} | {}} |
| <abstract\_func> | {func} |  |

Class Definition

|  |  |  |
| --- | --- | --- |
| <class\_def> | {type} |  |
| <enhance> | {enhances}  {ϵ}  {enhances, ϵ} | {{} |
| <enhance1> | {,}  {ϵ}  {, ϵ} | {{} |
| <class\_body> | {access\_modifier, passive, const, ID, func, [}  {ϵ}  {access\_modifier, passive, const, ID, func, [, ϵ} | {}} |
| <cb\_am\_ps\_cn> | {access\_modifier}  {passive}  {const}  {ID, func, [}  {access\_modifier, passive, const, ID, func, [} |  |
| <cb\_am\_ps\_cn1> | {ID, func, [}  {passive}  {const}  {ID, func, [, passive, const} |  |
| <am\_ps\_cn> | {access\_modifier}  {passive}  {const}  {ϵ}  {access\_modifier, passive, const, ϵ} | {ID, func, [} |
| <ps\_cn> | {passive}  {const}  {ϵ}  {passive, const, ϵ} | {ID, func, [} |
| <am\_cn> | {access\_modifier}  {const}  {ϵ}  {access\_modifier, const, ϵ} | {ID, func, [} |
| <am\_ps> | {access\_modifier}  {passive}  {ϵ}  {access\_modifier, passive, ϵ} | {ID, func, [} |
| <am> | {access\_modifier}  {ϵ}  {access\_modifier, ϵ} | {ID, func, [} |
| <cn> | {const}  {ϵ}  {const, ϵ} | {type, ID, func, [} |
| <ps> | {passive}  {ϵ}  {passive, ϵ} | {ID, func, [} |
| <cdecs> | {ID}  {func}  {[}  {ID, func, [} |  |
| <fn\_init\_dec> | {ID}  {constructor}  {ID, constructor} |  |
| <des\_dec> | {[} |  |
| <des\_dec1> | {,}  {]}  {,, ]} |  |

Language Structure

|  |  |  |
| --- | --- | --- |
| Rules | First Set | Follow Set |
| <lang> | {import, type, abstract, ID, func, [, $} | {$} |
| <defs> | {type}  {abstract}  {const}  {ID}  {[}  {type, abstract, const, ID, [} | {$} |
| <defs1> | {type, abstract, ID, const, [}  {ϵ}  {func}  {type, abstract, ID, func, const, [, ϵ} | {$} |
| <defs2> | {type, abstract, ID, const, [}  {func}  {ϵ}  {type, abstract, ID, func, const, [, ϵ} | {$} |
| <lang1> | {ID}  {main}  {ID, main} |  |
| <imports> | {import}  {ϵ}  {import, ϵ} | {const, type, abstract, ID, func, [, $} |
| <imports1> | {import}  {ϵ}  {import, ϵ} | {const, type, abstract, ID, func, [, $} |
| <import> | {import} |  |
| <main> | {main} |  |