\*\*\*\*\*\*\*\*\*\*\*

closureEval

class A { x[z] = a(1,2,3); }

Program

import

class A

Block

AssignmentStatement

ExpressionLValue

x

IdentExpression

z

ClosureEvalExpression

a

IntLitExpression

1

IntLitExpression

2

IntLitExpression

3

\*\*\*\*\*\*\*\*\*\*\*

def\_key\_value\_type1

class A {def C:@[string]; def S:@@[int:boolean];}

Program

import

class A

Block

VarDec

C

ListType

SimpleType

string

VarDec

S

KeyValueType

SimpleType

int

SimpleType

boolean

\*\*\*\*\*\*\*\*\*\*\*

def\_key\_value\_type1

class A {def C:@[string]; def S:@@[int:@@[string:boolean]];}

Program

import

class A

Block

VarDec

C

ListType

SimpleType

string

VarDec

S

KeyValueType

SimpleType

int

KeyValueType

SimpleType

string

SimpleType

boolean

\*\*\*\*\*\*\*\*\*\*\*

maplist1

class A { %4; }

Program

import

class A

Block

ExpressionStatement

IntLitExpression

4

\*\*\*\*\*\*\*\*\*\*\*

factor1

class A {def C={->x=y;};}

Program

import

class A

Block

ClosureDec

C

Closure

AssignmentStatement

IdentLValue

x

IdentExpression

y

\*\*\*\*\*\*\*\*\*\*\*

factor2

class A {def C={->x=y[z];}; def D={->x=y[1];};}

Program

import

class A

Block

ClosureDec

C

Closure

AssignmentStatement

IdentLValue

x

ListOrMapElemExpression

y

IdentExpression

z

ClosureDec

D

Closure

AssignmentStatement

IdentLValue

x

ListOrMapElemExpression

y

IntLitExpression

1

\*\*\*\*\*\*\*\*\*\*\*

factor3

class A {def C={->x=3;};}

Program

import

class A

Block

ClosureDec

C

Closure

AssignmentStatement

IdentLValue

x

IntLitExpression

3

\*\*\*\*\*\*\*\*\*\*\*

factor4

class A {def C={->x="hello";};}

Program

import

class A

Block

ClosureDec

C

Closure

AssignmentStatement

IdentLValue

x

StringLitExpression

hello

\*\*\*\*\*\*\*\*\*\*\*

factor5

class A {def C={->x=true; z = false;};}

Program

import

class A

Block

ClosureDec

C

Closure

AssignmentStatement

IdentLValue

x

BooleanLitExpression

true

AssignmentStatement

IdentLValue

z

BooleanLitExpression

false

\*\*\*\*\*\*\*\*\*\*\*

factor6

class A {def C={->x=-y; z = !y;};}

Program

import

class A

Block

ClosureDec

C

Closure

AssignmentStatement

IdentLValue

x

UnaryExpression

-

IdentExpression

y

AssignmentStatement

IdentLValue

z

UnaryExpression

!

IdentExpression

y

\*\*\*\*\*\*\*\*\*\*\*

def\_simple\_type1

class A {def B:int; def C:boolean; def S: string;}

Program

import

class A

Block

VarDec

B

SimpleType

int

VarDec

C

SimpleType

boolean

VarDec

S

SimpleType

string

\*\*\*\*\*\*\*\*\*\*\*

smallest

class A { }

Program

import

class A

Block

\*\*\*\*\*\*\*\*\*\*\*

emptyStatement

class A { ;;; }

Program

import

class A

Block

\*\*\*\*\*\*\*\*\*\*\*

list1

class A {

x = @[a,b,c];

y = @[d,e,f]+x;

}

Program

import

class A

Block

AssignmentStatement

IdentLValue

x

ListExpression

IdentExpression

a

IdentExpression

b

IdentExpression

c

AssignmentStatement

IdentLValue

y

BinaryExpression

ListExpression

IdentExpression

d

IdentExpression

e

IdentExpression

f

+

IdentExpression

x

\*\*\*\*\*\*\*\*\*\*\*

maplist1

class A { x = @@[x:y]; y = @@[x:y,4:5]; }

Program

import

class A

Block

AssignmentStatement

IdentLValue

x

MapListExpression

KeyValueExpression

IdentExpression

x

IdentExpression

y

AssignmentStatement

IdentLValue

y

MapListExpression

KeyValueExpression

IdentExpression

x

IdentExpression

y

KeyValueExpression

IntLitExpression

4

IntLitExpression

5

\*\*\*\*\*\*\*\*\*\*\*

expressions1

class A {def C={->x=x+1; z = 3-4-5;};}

Program

import

class A

Block

ClosureDec

C

Closure

AssignmentStatement

IdentLValue

x

BinaryExpression

IdentExpression

x

+

IntLitExpression

1

AssignmentStatement

IdentLValue

z

BinaryExpression

BinaryExpression

IntLitExpression

3

-

IntLitExpression

4

-

IntLitExpression

5

\*\*\*\*\*\*\*\*\*\*\*

expressions2

class A {def C={->x=x+1/2\*3--4; z = 3-4-5;};}

Program

import

class A

Block

ClosureDec

C

Closure

AssignmentStatement

IdentLValue

x

BinaryExpression

BinaryExpression

IdentExpression

x

+

BinaryExpression

BinaryExpression

IntLitExpression

1

/

IntLitExpression

2

\*

IntLitExpression

3

-

UnaryExpression

-

IntLitExpression

4

AssignmentStatement

IdentLValue

z

BinaryExpression

BinaryExpression

IntLitExpression

3

-

IntLitExpression

4

-

IntLitExpression

5

\*\*\*\*\*\*\*\*\*\*\*

expressions3

class A {def C={->x=x+1/2\*3--4; z = 3-4-5;};}

Program

import

class A

Block

ClosureDec

C

Closure

AssignmentStatement

IdentLValue

x

BinaryExpression

BinaryExpression

IdentExpression

x

+

BinaryExpression

BinaryExpression

IntLitExpression

1

/

IntLitExpression

2

\*

IntLitExpression

3

-

UnaryExpression

-

IntLitExpression

4

AssignmentStatement

IdentLValue

z

BinaryExpression

BinaryExpression

IntLitExpression

3

-

IntLitExpression

4

-

IntLitExpression

5

\*\*\*\*\*\*\*\*\*\*\*

expressions4

class A {x = a<<b; c = b>>z;}

Program

import

class A

Block

AssignmentStatement

IdentLValue

x

BinaryExpression

IdentExpression

a

<<

IdentExpression

b

AssignmentStatement

IdentLValue

c

BinaryExpression

IdentExpression

b

>>

IdentExpression

z

\*\*\*\*\*\*\*\*\*\*\*

def\_closure1

class A {def C={->};}

Program

import

class A

Block

ClosureDec

C

Closure

\*\*\*\*\*\*\*\*\*\*\*

def\_closure2

class A {def C={->}; def z:string;}

Program

import

class A

Block

ClosureDec

C

Closure

VarDec

z

SimpleType

string

\*\*\*\*\*\*\*\*\*\*\*

statements1

class A {x = y; z[1] = b; print a+b; print (x+y-z);}

Program

import

class A

Block

AssignmentStatement

IdentLValue

x

IdentExpression

y

AssignmentStatement

ExpressionLValue

z

IntLitExpression

1

IdentExpression

b

PrintStatement

BinaryExpression

IdentExpression

a

+

IdentExpression

b

PrintStatement

BinaryExpression

BinaryExpression

IdentExpression

x

+

IdentExpression

y

-

IdentExpression

z

\*\*\*\*\*\*\*\*\*\*\*

statements2

class A {

while (x) {};

while\* (1..4){};

while\*(x>0){}; }

Program

import

class A

Block

WhileStatement

IdentExpression

x

Block

WhileRangeStatement

RangeExpression

IntLitExpression

1

IntLitExpression

4

Block

WhileStarStatement

BinaryExpression

IdentExpression

x

>

IntLitExpression

0

Block

\*\*\*\*\*\*\*\*\*\*\*

statements3

class A {

if (x) {};

if (y){} else {};

if (x) {} else {if (z) {} else {};} ; }

Program

import

class A

Block

IfStatement

IdentExpression

x

Block

IfElseStatement

IdentExpression

y

Block

Block

IfElseStatement

IdentExpression

x

Block

Block

IfElseStatement

IdentExpression

z

Block

Block

\*\*\*\*\*\*\*\*\*\*\*

statements4

class A { %a(1,2,3); }

Program

import

class A

Block

ExpressionStatement

ClosureEvalExpression

a

IntLitExpression

1

IntLitExpression

2

IntLitExpression

3

\*\*\*\*\*\*\*\*\*\*\*

statements5

class A { x = a(1,2,3); }

Program

import

class A

Block

AssignmentStatement

IdentLValue

x

ClosureEvalExpression

a

IntLitExpression

1

IntLitExpression

2

IntLitExpression

3

\*\*\*\*\*\*\*\*\*\*\*

import1

import X; class A { }

Program

import

X

class A

Block

\*\*\*\*\*\*\*\*\*\*\*

import2

import X.Y.Z; import W.X.Y; class A { }

Program

import

X/Y/Z

W/X/Y

class A

Block