**<DEC>** : DT ID <init> <list>

<INIT> : = <init2>

<INIT> : e

<INIT2> : <const>

<INIT2> : ID <init>

<const> : int-const |str-const | bool-const | char-const | double-const

<list> : , ID <init><list>

<list> : ;

**<For\_st>** : for : <c1> <c2> ; <c3> : <body>

<c1> : <dec>

<c1> : <asgn\_st>

<c1> : ;

<c2> : <OE>

<c2> : e

<c3> : ID <X> <c4>

<c3> : inc\_dec ID <X>

<c3> : e

<c4> : <asgn\_op> <E>

<c4> : inc\_dec

**<Body>** : <SST>

<Body> **:** ( <MST> )

<Body> **:** ;

<SST> : <For\_st> | <while\_st> |<foreach\_st> |<if\_else> | <return\_st>| <ArrList\_st> | DT<SST03> |inc\_dec ID <X> ; |ID <SST01>

<SST01> : ID = new ID:<Arg\_st>:;

<SST01> : <Y> :<Argu\_st>: ;

<SST01> : <X> <SST02>;

<SST02> : <asgn\_op> <exp>

<SST02> : inc\_dec

<SST03> : ID <init> <list>

<SST03> : [] ID <list3> ;

<MST> : <SST><MST>

<MST> : e

**<obj\_dec> :** ID ID = new ID : <Arg\_st> : ;

<Argu\_st> : <ID\_const>

<ID\_const> : ID |<const>

<funct\_call> : ID <Y> : <Argu\_st> : ;

<Y> : .ID <Y>

<Y> : e

<Asgn\_st> : ID <X> <asgn\_op> <E>;

<Asgn\_op> : =

<Asgn\_op> : <CoP>

<CoP> : RO

<CoP> : AO

<X> : <Handle\_Array> <Handle\_Dot>

<X> : e

<Handle\_Array> : [int-const]

<Handle\_Array> : e

<Handle\_Dot> : .ID<X2>

<Handle\_Dot> : e

<X2> : <X>

<X2> : <Argu\_st>: <Handle\_Dot>

**<OE>** : <AE><OE2>

<OE2> : || <AE> <OE2>

<OE2> : e

<AE> : <RE> <AE2>

<AE2> : && <RE> <AE2>

<AE2> : e

<RE> : <E> <RE2>

<RE2> : RO <E> <RE2>

<RE2> : e

<E> : <T> <E2>

<E2> : PM <T> <E2>

<E2> : e

<T> : <F> <T2>

<T2> : MDM <F> <T2>

<T2> : e

<F> : <Th> ID <X>

<F> : <Const>

<F> : ( <OE> )

<F> : INC\_DEC ID

<F> : ! <F>

<Th> : this.

<Th> : Ɛ

**<foreach\_st>** : foreach : DT ID in <DS> : <body>

<DS> : <Arr\_St>

<DS> : <ArrList\_St>

**<While\_st>** : while : <OE> : <body>

**<if\_else>** : if : <OE> : <body> <elif\_st>

<elif\_st> : elif : <OE> : <body>

<elif\_st> : e

<elif\_st> : <else>

<else> : else <body>

<else> : e

**<funct\_st>**  : <static\_st> <DT\_Void> ID : <param> : <body>

<static\_st> : static

<static\_st> : e

<DT\_Void> : void

<DT\_Void> : DT

<param> : DT ID <multiParam>

<param> : e

<MultiParam> : , DT ID <multiParam>

<MultiParam> : e

**<Arr\_St>** : DT[] ID <list3> ;

<list3> : = <list2>

<list3> : e

<list2> : new DT [int]

<list2> : { <const> <multi\_value> }

<Multi\_Value> : ,<const> <multi\_value>

<Multi\_Value> : e

// <const> of same DT

**<ArrList\_St>** : ArrayList ID <list4> ;

<list4> : = new ArrayList() <list5>

<list4> : e

<list5> : { <const><multi\_value> }

<list5> : e

**<class\_st>** : <AccessModifier> <static\_abstract> class ID <inherit\_inter> <body2>

<AccessModifier> : AccessModifier

<AccessModifier> : e

<static\_abstract> : static

<static\_abstract> : abstract

<static\_abstract> : e

<Body2> : { <MST2> }

<SST2> : <body>

<SST2> : <funct\_st>

<SST2> : <class\_st>

<SST2> : e

<MST2> : <SST2><MST2>

<MST2> : e

<inherit\_inter> : <inherit\_st> <interface\_imple>

<inherit\_inter> : <interface\_imple>

**<inherit\_st>** : (: ) ID

**<inherit\_st>** : e

**<Poly\_st>** : <AccessModifier2> VO <funct\_st2>

<AccessModifier2> : public

<AccessModifier2> : protected

<funct\_st2> : <DT\_Void> ID : <param> : <body>

**<interface\_imple>** : <col\_comma> ID <interface\_imple>

**<interface\_imple>** : e

<col\_comma> : :

<col\_comma> : ,

**<Interface\_st>** : interface ID <Body3>

<Body3> : { <MST3> }

<MST3> : <SST3><MST3>

<MST3> : e

<SST3> : <funct\_St3>

<SST3> : e

<funct\_st3> : <DT\_void> ID : <params> : ;