|  |
| --- |
| 1- <start> 🡪 **{globalcurrScope = STobj.createScope() }**  <defs> <VI > main **{globalname = ts.vp}** (<NV> ) **{ Fnobj.insert (globalname, globalRetType, globalparalist , “global”, &fnstart) }** { **{ globalcurrScope = STobj.createScope() }** <MST >} **{ globalcurrScope = STobj.deleteScope() main scope ends }** $ **{ globalcurrScope = STobj.deleteScope() global scope ends}** | $  <VI > 🡪 void {**globalRetType = ts.vp}**  <NV> 🡪~ {**globalparalist =””}** | void {**globalparalist = ts.vp}**  <defs>🡪<class><defs> | ID **{globaltype = ts.vp }**  <defs1><defs > |DT **{globaltype = ts.vp }** ID **{globalname = ts.vp}** <defs2><defs> |static **{globalTM = ts .vp}**  <type> ( **{ globalcurrScope = STobj.createScope() }** <fn\_dec> | ~  <type**>** 🡪ID **{globaltype = ts.vp}** ID **{globalname = ts.vp}**| DT **{globaltype = ts.vp}**  ID **{globalname = ts.vp}**  <defs1 > 🡪<Ass\_st > | ID **{globalname = (\*curr)->vp }** <X>  <X> 🡪 <obj\_dec> | (<terminal>  <terminal> 🡪<fn\_dec> | <constructor\_dec>  <defs2> 🡪 <DT\_dec> | (**{globalcurrScope = STobj.createScope() }** <fn\_dec> |
| 2- <MST>🡪~ | <SST><MST> |
| 3- <SST>🡪<if\_else> | <while\_st> | <switch> | <for\_st> |<return> | continue;| break; | ID **{ globaltype= (\*curr)->vp }** <SST1> | DT **{globaltype = ts.vp}**  ID **{globalname = ts.vp}** <SST2>  <SST1> 🡪<array> <L1><L2>| ID **{ globalname= (\*curr)->vp }** <Xxx>  <L1>🡪 .ID <L1> | ~  <L2>🡪<fn\_call> | inc\_dec;|<Ass\_st>  <Xxx> 🡪<obj\_dec> | ( <constructor\_dec>  <SST2> 🡪 <DT\_dec> |
| 4- <return> 🡪 return <OE1> ;  <OE1> 🡪 <OE> | ~ |
| 5- <fn\_call> 🡪 <fn\_call 1>;  <fn\_call1>🡪 ( **{string fnname= globalrightType} {string operator = globaloperator} {string classname = STob.lookup(tempvar)}**<arg> **{Type = checkcompatibility(classname,fnname,globalparalist,operaotr)**  )  <arg> 🡪 <OE> **{globalparalist = Type + ”,”}** <arg1>| ~ **{globalparalist = “’}**  <arg1> 🡪 ,<OE> **{globalparalist +=Type +”,”}** <arg1>| ~ |
| 9- <inc\_dec> 🡪 <inc\_dec1>;  <inc\_dec1> 🡪 inc\_dec |
| 6- <DT\_dec> 🡪 <new\_arr> | <init>  <new\_arr> 🡪 [<OE>]<array2>  <array2>🡪= {<array3>} ; | ;  <array3> 🡪 <OE><array4>  <array4> 🡪 , <OE> <array4> | ~  <init> -->AOP <init2> | <list>  <init2> --> <OE> ;  <list> -->, **{STobj.insert(globalname , globaltype, globalcurrScope, &STstart) }** ID **{golbalname = (\*curr)->vp } { STobj.insert(globalname , globaltype, globalcurrScope, &STstart) }** <init3> <list> | ; **{ STobj.insert(globalname , globaltype, globalcurrScope, &STstart) }** |
| 7- <fn\_dec> --> <para>) **{fnobj.insert(globalname , globaltype , globalparalist ,{if(class)->globalclassname else->”Gobal”} , &fnstart) }** {<MST>} **{ globalcurrscope = STobj.deletescope() }**  <word1> --> virtual | static | const  <para> --> <def> <E> | void **{globalparalist = “void”} {globalcurrscope = STobj.createscope() }** | ~ **{globalparalist = “void” } {globalcurrscope = STobj.createscope()}**  <def> --> ID **{globaltype = (\*curr)->vp} { globalparalist += (\*curr->vp) }** ID **{globalname = (\*curr)->vp} { globalcurrscope = STobj.createscope() } {STobj.insert(globalname , globaltype , gobalcurrscope , &STstart)}** |DT **{globaltype = (\*curr)->vp}** **{globalparalist += (\*curr->vp) }** ID **{globalname = (\*curr)->vp}** **{ globalcurrscope = STobj.createscope() } {STobj.insert(globalname , globaltype , gobalcurrscope , &STstart)}**  <E> --> ,<def><E> | ~ |
| 8- <Ass\_st> --> <Ass\_st1><Xx> ;  <Ass\_st1> --> AOP **{string templeft = STobj.ookup(globaltype,STstart) }{globalOperator = (\*curr)->vp }** <OE> **{globalRightType = Type }{ Type = compatibilitycheck(globalleftType , globalRightType , “”,globalOperator }**  <Xx> --> <Ass\_st1><Xx> | ~  <array> --> [<OE>]| ~ |
| 10- <OE> --> <AE> <OE'>  <OE'> --> ||**{string globalleftype = Type } { globalOperator = (\*curr)->vp }**<AE>**{Type = compatibilitycheck (templeftType, globalRighttype , “ ” , globaloperator )}**<OE'> | ~  <AE> --> <RE> <AE'>  <AE'> --> && **{string globalleftype = Type } { globalOperator = (\*curr)->vp }** <RE>**{Type = compatibilitycheck (templeftType, globalRighttype , “ ” , globaloperator )}** <AE'> | ~  <RE> --> <PE><RE'>  <RE'> -->ROP **{string globalleftype = Type } { globalOperator = (\*curr)->vp }** <PE> **{Type = compatibilitycheck (templeftType, globalRighttype , “ ” , globaloperator )}** <RE'>|~  <PE> --> <ME><PE'>  <PE'> --> PM **{string globalleftype = Type } { globalOperator = (\*curr)->vp }** <ME> **{Type = compatibilitycheck (templeftType, globalRighttype , “ ” , globaloperator )}** <PE'>|~  <ME> --> <F><ME'>  <ME'> --> MDM **{string globalleftype = Type } { globalOperator = (\*curr)->vp }**  <F> **{Type = compatibilitycheck (templeftType, globalRighttype , “ ” , globaloperator )}** <ME'> | ~  <F> --> <CONST> **{ Type = (\*curr)->cp}** |(<OE>) |!<PE> |ID **{ tempvar = (\*curr)->vp} {globallefttype = STobj.loookup(tempvar, STstart}** <XOE1>  <XOE1> 🡪<fn\_call1> |<nt2> |~ **{** **Type = STobj.lookup(tempvar , STstart) }**  <nt2>🡪 .  **{globaloperator = (\*curr)->vp}** ID **{globalRightType = (\*curr)->vp** **} {Type = checkcompatibility (globalLefttype , globalRightType , “ ” , globaloperator )} {globalLeftType = Type}** <XOE1>|~ **{ Type = STobj.lookup(tempvar , STstart) }** |
| 11- <obj\_dec> --><array><new\_init> <list2>  <new\_init> --> AOP ID < new\_init > | ~ //remaining for semantic  <list2> --> ,**STobj.insert(globalname , globaltype, globalcurrScope, &STstart) }**ID **{golbalname =(\*curr)->vp} {STobj.insert(globalname , globaltype, globalcurrScope, &STstart) }** <array><new\_init><list2> | ; **{STobj.insert(globalname , globaltype, globalcurrScope, &STstart) }** |
| 12-<constructor\_dec> --> <arg>); |
| 12- <if\_else> --> if(<OE> **{if Type==CONST true else false}**){ **{globalcurrscope = STobj.createscope() }** <MST> **{globalcurrscope = STobj.deletescope()}** } <o\_else>  <o\_else> -->else { **{globalcurrscope = STobj.createscope()}** <MST> **{globalcurrscope = STobj.deletescope() }** }|~ |
| 13- <while\_st> --> while (<OE> **{if Type==CONST true else false}** ){ **{globalcurrscope = STobj.createscope()}** <MST> **{globalcurrscope = STobj.deletescope()}** } |
| 14- <switch> --> switch(<OE> **{if Type==CONST true else false}**  ){ **{globalcurrscope = STobj.createscope()}**  <case\_st><default> **{globalcurrscope = STobj.deletescope()}**  }  <case\_st> --> case <OE> **{if Type==CONST true else false}**  : {  **{globalcurrscope = STobj.createscope()}** <MST> **{globalcurrscope = STobj.deletescope()}** } <case>| ~  <default> --> default: {  **{globalcurrscope = STobj.createscope()}** <MST> **{globalcurrscope = STobj.deletescope()}**  } | ~ |
| 15- <class> --> class **{globaltype = (\*curr)->vp }**  ID **{globalclassname = (\*curr)->vp** **}** <chk\_inhrt> {  **{CDTRef = NULL} {DTobj.insertDT(globalclassname , globaltype , parent , CDTRef , &DTstart }** <class\_body> };**{DT\* DTtemp = DTobj.retAddress(globalclassname, DTstart) {DTtemp ->ref = CDTRef globalclassname = “Global”}**  <chk\_inhrt> -->: AM ID **{DT\* DTtemp = DTobj.retAddress((\*curr)->vp , DTstart) {parent = DTtemp}** | ~  <class\_body> -->ID **{ globaltype = (\*curr)->vp }** <X1> <class\_body> | DT **{ globaltype = (\*curr)->vp }** ID **{ globalname = (\*curr)->vp }** <X2><class\_body> | AM: **{globalAM = (\*curr)->vp}** <class\_body> | <word><type> (<fn\_dec> <class\_body>| ~  <word> --> virtual | const | static **{globalTM = (\*curr)->vp}**  <X1> --> ID **{ globalname = (\*curr)->vp }** <X3> | <intconstructor\_fn>  <X2> --> ( <fn\_dec> | <DT\_dec>  <X3> --> ( <fn\_dec> | <obj\_dec> |
| 16- <for\_st> -->for(**{createscope}** <C1><C2>;<C3>) { <MST> **{deletescope}** }  <C1> --> DT **{globaltype = (\*curr)->vp }** ID **{globalname = (\*curr)->vp } { globalflag = false }** <DT\_dec> | ID **{globaltype = (\*curr)->vp }** <Ass\_st>|;  <C2> --> <OE> **{if Type == int | float ||char ||bool ||string true else not compatible }** | ~  <C3> --> ID **{globaltype = (\*curr)->vp }** <X11> | inc\_dec | ~  <X11> --> <Ass\_st1> | inc\_dec |
| 17)<const> --> int | Float | string | bool |char |
| 18) < intconstructor \_fn> --> (  **{ DT\* DTtemp = DTobj.retAddress(globaltype , DTstart }** <para>) **{ Fnobj.insert (globalclassname, “void”, globalparalist , globalclassname , &fnstart }** {<MST>} **DONE** |