**C++ code structure**

|  |  |  |  |
| --- | --- | --- | --- |
| <Start> -> <defs> func execute () {<MST1>} | First(defs)= {AM,func,DT,ID,E} | {AM,func,DT,ID,E} | {$} |
| <defs> -> <class-st> <defs> | First(class-st) = AM | {AM,func,DT,ID,E} | {func} |
| <defs> -> <func-only> <defs> | First(func-only) = func |  |
| <defs> -> <DT> ID <dec1> <defs> | {DT ,ID} |  |
| <defs> -> E | E |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <func-only> -> func ID (<para>) : <RDT> {<MST1>} | Func | {func} | First(defs) = { AM,func,DT,ID,E} |
| <para> -> <DT’> ID <paras> | First(DT’)={DT,ID} | {DT,ID,E} | { ) } |
| <para> -> E | E |  |
| <paras> -> , <DT’> ID <paras> | , | {, , E} | Follow(para) = { ) } |
| <paras> -> E | E |  |
| <RDT> -> <DT’> | First(DT’)={DT,ID} | {DT,ID,void} | { { } |
| <RDT> -> void | Void |  |
| <DT’> -> DT <DT’’> | DT | {DT,ID} | {ID , { } |

|  |  |  |  |
| --- | --- | --- | --- |
| <DT’> -> ID <DT’’> | ID |  |  |
| <DT’’> -> [ ] <DT’’’> | [ | {[, E} | Follow (DT’) = {ID , { } |
| <DT’’> -> E | E |  |
| <DT’’’> -> [ ] | [ | {[,E} | Follow (DT’’) = {ID , { } |
| <DT’’’> -> E | E |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <MST1> -> <SST1> <MST1> | First(SST) = {while, if, break, skip, ret, DT, ID} | {while, if, break, skip, ret, DT, ID, E} | { } } |
| <MST1> -> E | E |  |
| <SST1> -> <while-st-1> | while | {while, if, break, skip, ret, DT, ID} | { while, if, break, skip, ret, DT, ID, } } |
| <SST1> -> <if-else-1> | if |
| <SST1> -><break-st>; | Break |
| <SST1> -> <continue-st>; | skip |
| <SST1> -> <return-st-1>; | ret |
| <SST1> -> <D-I-Fc-1> | First(D-I-Fc-1) = {DT, ID} |
| <D-I-Fc-1> -> DT ID <dec1> | DT | {DT, ID} | Follow(SST1) |
| <D-I-Fc-1> -> ID <D-S-A-FC-1> | ID |  |  |
| <D-S-A-FC-1> -> ID <dec1> | ID | {ID , . , ( , [ , = } | Follow(D-I-Fc-1) |
| <D-S-A-FC-1> -> <assign-fncall-1> | First(assign-fncall-1) = {. , ( , [ , = } |  |
| <assign-fncall-1> -> <f-n-1> | First(f-n-1) = {. , ( , [ } | {. , ( , [ , = } |  |
| <assign-fncall-1> -> =<exp1> | = |  |
| <f-n-1> -> . ID <A1> | . | {. , ( , [ } |  |
| <f-n-1> -> <arr-call-1><RID1> | [ |
| <f-n-1> -> (<argu1>)<T1> | ( |

|  |  |  |  |
| --- | --- | --- | --- |
| <A1> -> = <exp1> | = | {. , ( , [ , =} |  |
| <A1> -> <f-n-1> | First(f-n-1) = {. , ( , [ } |  |
| <T1> -> . ID <f-n-1> | . | {. , ;} |  |
| <T1> -> ; | ; |  |
| <RID1> -> .ID <f-n-1> | . | {. , = } |  |
| <RID1> -> = <exp1>; | = |  |
| <argu1> -> <exp1><argus1> | First(exp1) = {ID, INT , FLOAT , CHAR , STRING , ( , create} | {ID, INT , FLOAT , CHAR , STRING , ( , create, E} | { ) } |
| <argu1> -> E | E |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <argus1> -> , <exp1><argus1> | , | {, , E} | { ) } |
| <argus1> -> E | E |  |
|  | | | |
| <arr-call-1> -> [<exp1>][S1] | [ | { [ } | First(RID1) U first(O1’) – {E} U follow(O’) = {. , = ,MDM , PM, ROP, $ , ) , ] } |
| <S1> -> [<exp1>] | [ | { [ , E} | Follow(arr-call-1) = {. , = ,MDM , PM, ROP, $ , ) , ] } |
| <S1> -> E | E |

|  |  |  |  |
| --- | --- | --- | --- |
| <dec1>-> [ ]<M1> | [ | { = , [, ;, ,} | { while, if, break, skip, ret, DT, ID, } } |
| <dec1> -> <init1><list1> | First(init1) = { = , E} |  |  |
| <list1> -> ; | ; | { ; , ,} |  |
| <list1> -> ,ID<init1><list1> | , |  |  |
| <init1> -> = <init1’> | = | { = , E} | { ; , ,} |
| <init1> -> E | E |  |  |
| <init1’> -> ID <init1’’> | ID | {ID, INT , FLOAT , CHAR , STRING , ( , create } |  |
| <init1’> - > <const><H1’><G1’><exp1’> | {INT, FLOAT, CHAR, STRING} |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <init1’> - > (<exp1>)<H1’><G1’><exp1’> | ( |  |  |
| <init1’> -> <obj-dec-1> <O1’><H1’><G1’><exp1’> | Create |  |  |
| <init1’’> -> <O1> <H1'><G1'><exp1'> | First(O1) = {. , [, (, E} | {. , [ , ( , = , E} | { ; , ,} |
| <init1’’> - > <init1> | First(init1) = { = , E} |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <M1> -> [] <M1’> | [ | {= , [, E} | { while, if, break, skip, ret, DT, ID, } } |
| <M1> -> <arr-init1><arr-list1> | First(arr-init1) = {= , E} |  |  |
| <M1’> -> <arr-init2f><arr-list1> | First(arr-init2f)= { = ,E} | { = ,E} | { while, if, break, skip, ret, DT, ID, } } |
| <arr-list1> -> ; | ; | {; , , } |  |
| <arr-list1> ->, ID [ ] <M1> | , |  |  |
| <arr-init1> -> = <arr-init1’> | = | {= , E} | { ; , , } |
| <arr-init1> -> E | E |  |  |
| <arr-init1’> -> {<element1>} | { | {DT , { , ID} |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <arr-init1’> - > <DT> [<exp1>] | {DT,ID} |  |  |
| <element1>-> <const><elements1> | First(const) = { INT , FLOAT , CHAR , STRING , } | = {INT , FLOAT , CHAR , STRING , E} | { } } |
| <element1>-> E | E |  |  |
| <elements1> -> , <const><elements1> | , | { , , E} | { } } |
| <elements1> ->E | E |  |  |
| <arr-init2f> -> = <arr-init2f’> | = | { = ,E} | { ; , , } |
| <arr-init2f> ->E | E |  |  |
| <arr-init2f’> -> {<element2f>} | { | {DT, { , ID} |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <arr-init2f’> -> <DT>[<exp1>] [<exp1>] | {DT , ID} |  |  |
| <element2f>-> {<element1>}<2df> | { | { E, { } | { } } |
| < element2f>-> -> E | E |  |  |
| <2df> -> , {<element1>}<2df> | , | {, , E} | { } } |
| <2df> -> E | E |  | |
|  |  |  | |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| <exp1> -> <G1> <exp1’> | First(G1) = {ID, INT , FLOAT , CHAR , STRING , ( , create} | {ID, INT , FLOAT , CHAR , STRING , ( , create} | { ) , ] } |
| <exp1’> -> ROP <G1><exp1’> | ROP | {ROP,E} | { ) , ] } |
| <exp1’> -> E | E |  |  |
| <G1> -> <H1> <G1’> | First(H1) = {ID, INT , FLOAT , CHAR , STRING , ( , create} | {ID, INT , FLOAT , CHAR , STRING , ( , create} | {ROP , ) , ] } |
| <G1’> -> PM <H1><G1’> | PM | {PM,E} | {ROP, ) , ] } |
| <G1’> -> E | E |  |  |
| <H1> -> <F1><H1’> | First(F1) = {ID, INT , FLOAT , CHAR , STRING , ( , create} | {ID, INT , FLOAT , CHAR , STRING , ( , create} | {PM, ROP, ) , ] } |
| <H1’>-> MDM <F1><H1’> | MDM | {MDM , E} | {PM, ROP , ) , ] } |

|  |  |  |  |
| --- | --- | --- | --- |
| <H1’> -> E | E |  |  |
| <F1> -> ID <O1> | ID | {ID, INT , FLOAT , CHAR , STRING , ( , create} | {MDM , PM, ROP , ) , ] } |
| <F1> -> <const> | INT , FLOAT , CHAR , STRING |  |  |
| <F1> -> (<exp1>) | ( |  |  |
| <F1> -> <obj-dec-1><O1’> | Create |  |  |
| <O1> -> <arr-call-1><O1’> | [ | {. , [, (, E} | Follow(F1) = {MDM , PM, ROP, ) , ] } |
| <O1> -> (<argu1>)<O1’> | ( |  |  |
| <O1> -> <O1’> | First(O1’) = {. , E} |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <O1’> -> .ID<O1> / E | {. , E} | {. , E} | Follow(O1) = {MDM , PM, ROP, ) , ] } |

|  |  |  |  |
| --- | --- | --- | --- |
| <while-st-1> -> while (<exp1>) <body1> | While | {while} |  |
| <body1> - > ; | ; | {while, if, break, skip, ret, DT, ID , ; , { } |  |
| <body1> -> <SST1> | First(SST) = {while, if, break, skip, ret, DT, ID} |  |  |
| <body1> -> {<MST1>} | { |  |  |
|  |  |  |  |
| <if-else-1> -> if (<exp1>)<body1><else1> | If | {if} | Follow(SST1) |
| <else1> -> else <body1> | Else | {else , E} | Follow(SST) |
| <else1> -> E | E |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <return-st-1> -> ret <case1> | Ret | {ret} |  |
| <case1> -> <exp1> | First(exp) = {ID, INT , FLOAT , CHAR , STRING , ( , create} | {ID, INT , FLOAT , CHAR , STRING , ( , create ,null} |  |
| <case1> -> null | Null |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <obj-dec-1> -> create ID (<arug1>) | create | {create} |  |

OOP

|  |  |  |  |
| --- | --- | --- | --- |
| <SST> - > <while-st> | while | {while, if, break, skip, ret, super, self, DT ,ID} |  |
| <SST> - > <if-else-st> | if |  |  |
| <SST> - > <break-st> | break |  |  |
| <SST> - > <continue-st> | skip |  |  |
| <SST> - > <return-st> | ret |  |  |
| <SST> - > super.ID<assign-fn-call> | super |  |  |
| <SST> - > self.ID<assign-fn-call> | self |  |  |
| <SST> -> <D-I-Fc> | DT, ID |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <MST> -> <SST> <MST> | {while, if, break, skip, ret, super, self, DT ,ID} | {while, if, break, skip, ret, super, self, DT ,ID,E} | { } } |
| <MST> -> E | E |  |  |
| <D-I-Fc> ->DT ID <dec> | DT | {DT,ID} |  |
| <D-I-Fc> ->ID <D-S-A-Fc> | ID |  |  |
| <D-S-A-Fc> -> ID <dec> | ID | {ID , . , ( , [ , = } |  |
| <D-S-A-Fc> -> <assign-fn-call> | First(assign-fn-call)= {. , ( , [ , = } |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <assign-fncall> -> <f-n> | First(f-n) = {. , ( , [ } | {. , ( , [ , = } |  |
| <assign-fncall> -> =<exp> | = |  |
| <f-n> -> . ID <A> | . | {. , ( , [ } |  |
| <f-n> -> <arr-call> <RID> | [ |
| <f-n> -> (<argu>)<T> | ( |

|  |  |  |  |
| --- | --- | --- | --- |
| <A> -> = <exp> | = | {. , ( , [ , =} |  |
| <A> -> <f-n> | First(f-n) = {. , ( , [ } |  |
| <T> -> . ID <f-n> | . | {. , ;} |  |
| <T> -> ; | ; |  |
| <RID> -> .ID <f-n> | . | {. , = } |  |
| <RID> -> = <exp>; | = |  |
| <argu> -> <exp><argus> | First(exp) = {self, super, ID, INT , FLOAT , CHAR , STRING , ( , create} | { self, super, ID, INT , FLOAT , CHAR , STRING , ( , create, E} | {)} |
| <argu> -> E | E |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <argus> -> , <exp><argus> | , | {, , E} | { ) } |
| <argus> -> E | E |  |
|  | | | |
| <arr-call-> -> [<exp>][S] | [ | { [ } |  |
| <S> -> [<exp>] | [ | { [ , E} | {. , = ,MDM , PM, ROP, $ , ) , ] } |
| <S> -> E | E |

|  |  |  |  |
| --- | --- | --- | --- |
| <exp> -> <G> <exp’> | First(G) = { self, super , ID, INT , FLOAT , CHAR , STRING , ( , create} | { self, super , ID, INT , FLOAT , CHAR , STRING , ( , create} | { $ , ) , ] } |
| <exp’> -> ROP <G><exp’> | ROP | {ROP,E} | { ) , ] } |
| <exp’> -> E | E |  |  |
| <G> -> <H> <G’> | First(H) = { self, super , ID, INT , FLOAT , CHAR , STRING , ( , create} | { self, super , ID, INT , FLOAT , CHAR , STRING , ( , create} | {ROP, ) , ] } |
| <G’> -> PM <H><G’> | PM | {PM,E} | {ROP , ) , ] } |
| <G’> -> E | E |  |  |
| <H> -> <F><H’> | First(F) = { self, super , ID, INT , FLOAT , CHAR , STRING , ( , create} | { self, super , ID, INT , FLOAT , CHAR , STRING , ( , create} | {PM, ROP, ) , ] } |
| <H’>-> MDM <F><H’> | MDM | {MDM , E} | {PM, ROP, ) , ] } |

|  |  |  |  |
| --- | --- | --- | --- |
| <H’> -> E | E |  |  |
| <F> -><TS> ID <O> | ID | {self, super , ID, INT , FLOAT , CHAR , STRING , ( , create} | {MDM , PM, ROP, ) , ] } |
| <F> -> <const> | INT , FLOAT , CHAR , STRING |  |  |
| <F> -> (<exp>) | ( |  |  |
| <F> -> <obj-dec><O’> | Create |  |  |
| <O> -> <arr-call><O’> | [ | {. , [, (, E} | Follow(F1) = {MDM , PM, ROP, ) , ] } |
| <O> -> (<argu>)<O’> | ( |  |  |
| <O> -> <O’> | First(O’) = {. , E} |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <O1’> -> .ID<O1> / E | {. , E} | {. , E} | Follow(O1) = {MDM , PM, ROP, ) , ] } |

|  |  |  |  |
| --- | --- | --- | --- |
| <TS> super. | super | {super, self , E} | {ID} |
| <TS> self. | self |  |  |
| <TS> E | E |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <fn-st> -> func AM <R> | func | {func} |  |
| <R> ->< s-fn-st> | Static , const ,ID | { Static , const ,ID, abstract } |  |
| <R> -> abstract <D> ; | abstract |  |  |
|  |  |  |  |
| < s-fn-st> -> static <C> <D> <E> | static | {static, const, ID} |  |
| < s-fn-st> -> const <D><E> | const |  |  |
| <s-fn-st> -> <D><D’> | ID |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <C> -> const / E | Const , E | {const ,E} | {ID} |
| <D> -> ID (<para>) | ID | {ID} |  |
| <D’> -> <E> | : | {: , { } |  |
| <D’> -> {<const-body>} | { |  |  |
| <E> -> : <RDT> {<MST>} | : | {:} |  |
| <const-body> -> <super> <MST> | super | {super} |  |
| <super> -> super(<argu>); | super | {super} |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <class-st> -> AM <P> | AM | {AM} |  |
| <P> -> abstract<S>{<A-body>} | abstarct | {abstract, const,class} |  |
| <P> -> const <S>{<s-body>} | const |  |  |
| <P> -> <S>{<s-body>} | class |  |  |
| <S> ->class ID <K> | class | {class} |  |
| <K> -> childOf ID<K’> | childOf | {childOf, E} | Follow(S) = { { } |
| <K> -> E | E |  |  |
| <K’> -> ,ID <K’> | , | {E , ,} | Follow(K) = { { } |

|  |  |  |  |
| --- | --- | --- | --- |
| <K’> -> E | E |  |  |
| <s-body> -> AM <B><C><DT’> ID <dec><s-body> | AM | {AM , func, E} | { } } |
| <s-body> -> func AM<s-fn-st><s-body> | Func |  |  |
| <s-body> -> E | E |  |  |
| <A-body> -> AM <B><C> <DT’> ID <dec> <A-body> | AM | {AM , func, E} | { } } |
| <A-body> -> func AM <A-body’><A-body> | Func |  |  |
| <A-body> -> E | E |  |  |
| <A-body’> -> abstract <D>; | Abstract | {abstract , static , const, ID} |  |

|  |  |  |  |
| --- | --- | --- | --- |
| <A-body’>-> static <C><D><E> | Static |  |  |
| <A-body’> -> const <D><E> | const |  |  |
| <A-body’>-> <D><E> | ID |  |  |
| <B> -> static / E | {static , E} | {static , E} | {const , DT,ID} |
|  |  |  |  |
| <argu> -> <exp><argus> | First(exp) = { self, super , ID, INT , FLOAT , CHAR , STRING , ( , create} | { self, super , ID, INT , FLOAT , CHAR , STRING , ( , create ,E} | { ) } |
| <argu>-> E |  |  |  |
| <argus> -> , <exp><argus> / E | {, , E} | {, , E} | { )} |