*Program -> {Req} int main(){Declarations Statements}*

*Req -> Require(‘Type’);*

*Declarations -> {Declaration}*

*Declaration -> Type | DS Identifier{, Identifier}*

*Type -> int | bool | float | char | string*

*DS -> list(Type) | array(Type) | tree(Type) | stack(Type) | queue(Type) | node(Type)*

*Statements -> {Statement}*

*Statement -> ; | Block | Assignment | IfStatement | WhileStatement | SortingStatement | OutputStatement | InputStatement | PushStatement*

*Block -> {Statements}*

*Assignment -> Identifier[[Expression]]=Expression;*

*IfStatement -> if(Expression)Statement[else Statement ]*

*WhileStatement -> while(Expression)Statement*

*SortingStatement -> sorting(Identifier);*

*OutputStatement -> output(“String” | Identifier{ + “String” | Identifier});*

*InputStatement -> Identifier = Input(“String”);*

*PushStatement -> Identifier.push(Expression);*

*Expression -> Conjunction{ || Conjunction}*

*Conjunction -> Equality{ && Equality}*

*Equality -> Relation[EquOp Relation]*

*EquOP -> == | !=*

*Relation -> Addition[RelOp Addition]*

*RelOp -> < | <= | > | >=*

*Addition -> Term{ AddOp Term}*

*AddOp -> + | -*

*Term -> Factor{MulOp Factor}*

*MulOp -> \* | / | %*

*Factor -> [UnaryOp]Primary*

*UnaryOp -> - | !*

*Primary -> Max(Expression) | Min(Expression) | Identifier[ [Expression] ] | Literal | (Expression) | Type (Expression) | DS | Identifier.pop() | Identifier.tree(Type {, Type} ) | Identifier{. Right() | Left() | Head() | Tail() | Root() | Parent() }*

*Identifier -> Letter{ Letter | Digit}*

*Letter -> a | b | … | Y | Z*

*Digit -> 0 | 1 | 2 | … | 9*

*Literal -> Integer | Boolean | Float | Char | String*

*Integer -> Digit{ Digit }*

*Boolean -> true | false*

*Float -> Integer.Integer*

*Char -> ‘ASCII Char’*

*String -> char{char}*