**Compilers Phase2 Part1**

**Prepared by:**

|  |  |  |
| --- | --- | --- |
| **Name** | **ID** | **Group** |
| Hazem Mohamed Taha | 20140097 | CSIS4 |
| Noura Mohi | 20140293 | CSIS4 |
| Mostafa Mohamed Mahmoud | 20140275 | CSIS3 |
| Mostafa Taher | 20140267 | CSIS3 |

**Github link:** https://github.com/Mostafa1201/Compiler

- Goal::= MainClass ( ClassDeclaration )\* <EOF>

- MainClass::= "class" Identifier "{" "public" "static" "void" "main" "(" "String" "[" "]" Identifier ")" "{" Statement "}" "}"

- ClassDeclaration::= "class" Identifier ( "extends" Identifier )? "{" ( VarDeclaration )\* (ConstructorDeclaration )\* ( MethodDeclaration )\* "}"

- VarDeclaration::= Type Identifier ";"

- ConstructorDeclaration::= Identifier "(" ( Type Identifier ( "," Type Identifier )\*)? ")" "{" ( VarDeclaration )\* ( Statement )\* "}"

- MethodDeclaration::= ("public" | "private" | "protected") Type Identifier  
"(" ( Type Identifier ( "," Type Identifier )\*)? ")" "{" ( VarDeclaration )\* ( Statement )\* "return" Expression ";" "}"

- Type::= "int" TypeDash  
| "float" TypeDash  
| "String" TypeDash  
| "char" TypeDash  
| "boolean" TypeDash

* TypeDash ::= "[" "]"| lamda

- Statement ::= "{" ( Statement )\* "}"  
| "if" "(" Expression ")" Statement StatementDash   
| "while" "(" Expression ")" Statement  
| "System.out.println" "(" Expression ")" ";"  
| Identifier StatementIdentifierDash

* StatementDash ::= "else" Statement | lamda
* StatementIdentifierDash ::= "=" Expression ";" | "[" Expression "]" "=" Expression ";"

- Expression ::=

<INTEGER\_LITERAL> ExpressionDash  
| <FLOAT\_LITERAL> ExpressionDash  
| "true" ExpressionDash

| "false" ExpressionDash  
| Identifier ExpressionDash  
| "this" ExpressionDash  
| "new" ExpressionNewDash  
| "!" Expression ExpressionDash  
| "(" Expression ")" ExpressionDash

* ExpressionDash ::=

( "&&" | "||" | "==" | "!=" | ">" | "<" |"<=" | ">="| "+" | "-" | "\*" | "/" ) Expression ExpressionDash

|"[" Expression "]" ExpressionDash

| "." ExpressionDotDash| lamda

* ExpressionNewDash ::= ("int" | "float" | "String" | "char" | "boolean") "[" Expression "]" ExpressionDash | Identifier "(" (Expression ( "," Expression)\*)? ")" ExpressionDash
* ExpressionDotDash ::= "length" ExpressionDash | Identifier "(" ( Expression ( "," Expression )\* )? ")" ExpressionDash