Program 🡪 Program’ Main\_Function

Program’ 🡪 Function\_Statements | [**Ɛ**](https://en.wikipedia.org/wiki/Latin_epsilon)



Fuction\_Statements 🡪 Function\_Statement Fun\_Ss

Fun\_Ss 🡪 Function\_Statement Fun\_Ss | [Ɛ](https://en.wikipedia.org/wiki/Latin_epsilon)

1. Function\_Statement 🡪 Function\_DeclerationFunction\_Body
2. Main\_Function 🡪 Datatype main () Function\_Body
3. Function\_Decleration 🡪 Datatype Identifier(Argument |[**Ɛ**](https://en.wikipedia.org/wiki/Latin_epsilon))
4. Function\_Body 🡪 { **Statements** Return\_Statement}
5. Datatype 🡪 int | float | string

Argument 🡪 Datatype identifier Args

Args 🡪 , Datatype identifier Args | [Ɛ](https://en.wikipedia.org/wiki/Latin_epsilon)



Statements → Statement State

State 🡪 statements | [Ɛ](https://en.wikipedia.org/wiki/Latin_epsilon)

1. Statement 🡪 Write\_Statement | Read\_Statement | If\_Statement | Assignment\_Statement | Declaration\_Statement | Return\_Statement | Repeat\_Statement
2. Write\_Statement 🡪 write (Expression |endl) ;
3. Read\_Statement 🡪 read identifier ;
4. If\_Statement 🡪 if Condition\_Statement then Statements (Else\_If\_Statement | Else\_Statement|end)
5. Assignment\_Statement 🡪identifier := Expression

**Decleration\_statement --> Datatype identifier | Assignment\_statement Additional\_decleration; Additional\_decleration --> , identifier | Assignment\_statement Additional\_decleration | Ɛ**

OR

**Declaration\_Statement** 🡪 **Datatype Assignment\_Statements**  ;

**Assignment\_Statements**🡪**Assignment\_Statement   Assigns\_coma | Identifiers Idens\_coma**

**Assigns\_coma** 🡪**,  Assignment\_Statements | Ɛ**

**Identifiers**🡪**Identifier Idens\_coma**

**Idens\_coma 🡪 , Assignment\_Statements |**[Ɛ](https://en.wikipedia.org/wiki/Latin_epsilon)

1. Return\_Statement 🡪 return Expression ;
2. Repeat\_Statement 🡪 repeat Statements until Condition\_Statement
3. Condition\_Statement 🡪 condition more\_cond
4. more\_cond 🡪 Boolean\_op condition\_statems | [Ɛ](https://en.wikipedia.org/wiki/Latin_epsilon)
5. Else\_If\_Statement 🡪 elseif Condition\_Statement then

Statements (Else\_If\_Statement | Else\_Statement|end)

1. Else\_Statement 🡪 else Statements end
2. Condition 🡪 Identifier Condition\_Operator Term
3. Expression 🡪 string | Term | Equation
4. Term 🡪 number | identifier | Function\_Call
5. Equation 🡪 Equation AddOp Equation | Ter

x --> id | fun | num | ( y )

y --> x add | mu x

equation --> y

OR

Equation 🡪 Ter eq ‘

Eq ‘ 🡪 AddOp Equation Eq ‘| [Ɛ](https://en.wikipedia.org/wiki/Latin_epsilon)

1. Ter 🡪 Ter MulOp Ter | Factor

Ter 🡪 Factor Ter ‘

Ter ‘ 🡪 MulOp Ter Ter ‘| [Ɛ](https://en.wikipedia.org/wiki/Latin_epsilon)

1. Factor 🡪 (Equation) | Term
2. AddOp 🡪 + | -
3. 30. Condition\_Operator 🡪 > | < | = | <>
4. 32. MulOp 🡪 \* | /
5. 33.Boolean\_op 🡪 && | ||
6. Function\_Call 🡪 Identifier (Term paramter| [Ɛ](https://en.wikipedia.org/wiki/Latin_epsilon))

paramter 🡪 , Term parameter | [Ɛ](https://en.wikipedia.org/wiki/Latin_epsilon)