

SYNTAX ANALYZER

- START:

$\langle s \rangle \rightarrow \langle C_I \rangle \text{class Main \{ Main() \{ \langle non_func_MST \rangle \} \} \langle C_I \rangle}$
 $\langle C_I \rangle \rightarrow \langle class \rangle \langle C_I \rangle \mid \langle interface \rangle \langle C_I \rangle \mid \epsilon$

- CLASS:

$\langle class \rangle \rightarrow \text{AM} \langle sealed \rangle \text{ class ID } \langle inherit_interface \rangle \{ \langle non_class_MST \rangle \}$
 $\langle sealed \rangle \rightarrow \text{sealed} \mid \epsilon$
 $\langle inherit_interface \rangle \rightarrow : \text{ID } \langle multi_interface \rangle \mid \epsilon$
 $\langle multi_interface \rangle \rightarrow , \text{ID} \langle multi_interface \rangle \mid \epsilon$

- INTERFACE:

$\langle interface \rangle \rightarrow \text{interface ID } \{ \langle func_interface \rangle \}$
 $\langle func_interface \rangle \rightarrow \langle ret_T \rangle \text{ ID } (\langle parameter \rangle); \langle func_interface \rangle \mid \epsilon$

- DECLARATION:

$\langle list \rangle \rightarrow \langle list1 \rangle \mid = \langle L' \rangle$
 $\langle list1 \rangle \rightarrow ; \mid , \text{ID} \langle list \rangle$
 $\langle L' \rangle \rightarrow \langle OE \rangle \langle list \rangle$

- ARRAY:

$\langle A' \rangle \rightarrow , \langle OE \rangle] = \{ \langle 2D_cond \rangle \}; \mid] = \{ \langle condition \rangle \};$
 $\langle 2D_cond \rangle \rightarrow \{ \langle condition \rangle \} \langle 2D_cond1 \rangle$
 $\langle 2D_cond1 \rangle \rightarrow , \langle 2D_cond \rangle \mid \epsilon$

- OBJECT:

$\langle object \rangle \rightarrow \text{ID ID} = \text{new ID} (\langle condition \rangle);$

- FUNCTION:

- FUNCTION DEFINITION:

$\langle ret_T \rangle \rightarrow \text{void} \mid \text{DT} \langle const_array \rangle$

$\langle const_array \rangle \rightarrow \langle C' \rangle \mid \epsilon$

$\langle C' \rangle \rightarrow [\langle C'' \rangle$

$\langle C'' \rangle \rightarrow ,] \mid]$

- **LOOPS:**

- **FOR LOOP:**

- $\langle \text{for_loop} \rangle \rightarrow \text{for}(\langle c1 \rangle \langle c2 \rangle ; \langle c3 \rangle) \{ \langle \text{non_func_MST} \rangle \}$

- $\langle c1 \rangle \rightarrow \text{DT ID} \langle \text{list} \rangle \mid \langle c3 \rangle ;$

- $\langle c2 \rangle \rightarrow \langle \text{OE} \rangle \mid \epsilon$

- $\langle c3 \rangle \rightarrow \langle \text{Asg_Op_loop} \rangle \mid \epsilon$

- **WHILE LOOP:**

- $\langle \text{while_loop} \rangle \rightarrow \text{while}(\langle \text{OE} \rangle) \{ \langle \text{non_func_MST} \rangle \}$

- **Assignment Operator For Loop**

- $\langle \text{Asg_Op_loop} \rangle \rightarrow \text{this.} \langle \text{Asg_Op}' \rangle \mid \langle \text{Asg_Op}' \rangle \mid \text{DI ID} \langle \text{ID_array} \rangle$
 $\langle \text{AsgOp_Comma} \rangle$

- $\langle \text{Asg_Op}' \rangle \rightarrow \text{ID} \langle \text{ID} \rangle \langle \text{ID_array} \rangle \langle \text{Asg_Op}'' \rangle$

- $\langle \text{Asg_Op}'' \rangle \rightarrow , \langle \text{Asg_Op_loop} \rangle \langle \text{Asg_Op_loop1} \rangle \mid \langle \text{Asg_Op_loop1} \rangle \mid \text{DI}$
 $\langle \text{AsgOp_Comma} \rangle$

- $\langle \text{Asg_Op_loop1} \rangle \rightarrow \langle \text{AsgOp_Equal} \rangle \langle \text{OE} \rangle \langle \text{Asg_Op_loop1} \rangle \langle \text{AsgOp_Comma} \rangle \mid \epsilon$

- $\langle \text{AsgOp_Comma} \rangle \rightarrow , \langle \text{Asg_Op_loop} \rangle \mid \epsilon$

- $\langle \text{AsgOp_Equal} \rangle \rightarrow \text{AsgOp} \mid \text{equal}$

- **IF-ELSE:**

- $\langle \text{if-else} \rangle \rightarrow \text{if}(\langle \text{OE} \rangle) \{ \langle \text{non_func_MST} \rangle \} \langle \text{else} \rangle$

- $\langle \text{else} \rangle \rightarrow \text{else} \langle \text{else_if} \rangle \mid \epsilon$

- $\langle \text{else_if} \rangle \rightarrow \{ \langle \text{non_func_MST} \rangle \} \mid \langle \text{if-else} \rangle$

- **TRY, CATCH, FINALLY:**

- $\langle \text{TCF} \rangle \rightarrow \text{try} \{ \langle \text{non_func_MST} \rangle \} \text{catch} \{ \langle \text{non_func_MST} \rangle \} \text{finally} \{ \langle \text{non_func_MST} \rangle \}$

- **OBJECT:**

- $\langle \text{object} \rangle \rightarrow \text{ID ID} = \text{new ID}(\langle \text{condition} \rangle);$

- **EXPRESSION:**

- $\langle \text{OE} \rangle \rightarrow \langle \text{A_OE} \rangle \langle \text{OE}' \rangle$

$\langle OE' \rangle \rightarrow OR \langle A_{OE} \rangle \langle OE' \rangle \mid \epsilon$
 $\langle A_{OE} \rangle \rightarrow \langle R_{OE} \rangle \langle A_{OE'} \rangle$
 $\langle A_{OE'} \rangle \rightarrow AND \langle R_{OE} \rangle \langle A_{OE'} \rangle \mid \epsilon$
 $\langle R_{OE} \rangle \rightarrow \langle E_{OE} \rangle \langle R_{OE'} \rangle$
 $\langle R_{OE'} \rangle \rightarrow ROP \langle E_{OE} \rangle \langle R_{OE'} \rangle \mid \epsilon$
 $\langle E_{OE} \rangle \rightarrow \langle T_{OE} \rangle \langle E_{OE'} \rangle$
 $\langle E_{OE'} \rangle \rightarrow PM \langle T_{OE} \rangle \langle E_{OE'} \rangle \mid \epsilon$
 $\langle T_{OE} \rangle \rightarrow \langle F_{OE} \rangle \langle T_{OE'} \rangle$
 $\langle T_{OE'} \rangle \rightarrow MDM \langle F_{OE} \rangle \langle T_{OE'} \rangle \mid \epsilon$
 $\langle F_{OE} \rangle \rightarrow \langle const \rangle \mid !(\langle OE \rangle) \mid DI \ ID \langle ID_array \rangle \mid this. \langle F' \rangle \mid ID \langle F'' \rangle$
 $\langle F' \rangle \rightarrow ID \langle F'' \rangle$
 $\langle F'' \rangle \rightarrow . \langle F' \rangle \mid [\langle OE \rangle \langle arr_call \rangle] \ \langle Dec_inc \rangle \mid DI \mid \epsilon$

- **GENERALIZED NON-TERMINAL:**

$\langle Dec_inc \rangle \rightarrow DI \mid \epsilon$
 $\langle ID_array \rangle \rightarrow [\langle OE \rangle \langle arr_call \rangle] \mid \epsilon$
 $\langle arr_call \rangle \rightarrow , \langle OE \rangle \mid \epsilon$
 $\langle return \rangle \rightarrow return \ \langle return' \rangle$
 $\langle return' \rangle \rightarrow \langle OE \rangle ; \mid \epsilon$

$\langle base \rangle \rightarrow :base(\langle argument \rangle) \mid \epsilon$
 $\langle condition \rangle \rightarrow \langle argument \rangle \mid \epsilon$
 $\langle argument \rangle \rightarrow \langle OE \rangle \langle a \rangle$
 $\langle a \rangle \rightarrow , \langle argument \rangle \mid \epsilon$

$\langle parameter \rangle \rightarrow DT \ ID \langle add_para \rangle \mid \epsilon$
 $\langle add_para \rangle \rightarrow , \langle parameter \rangle \mid \epsilon$

$\langle ID \rangle \rightarrow .ID \langle ID \rangle \mid \epsilon$

$\langle non_func_MST \rangle \rightarrow \langle SST_F \rangle \langle non_func_MST \rangle \mid \epsilon$
 $\langle SST_F \rangle \rightarrow DT \ ID \ \langle SST_F'' \rangle \mid this. \ \langle SST_F1' \rangle \mid ID \ \langle SST_F1_obj \rangle \mid \langle for_loop \rangle$
 $\mid \langle while_loop \rangle \mid \langle if_else \rangle \mid \langle TCF \rangle \mid DI \ ID \ \langle ID_array \rangle \mid break; \mid$
 $continue; \mid \langle return \rangle$

$\langle SST_F'' \rangle \rightarrow \langle list \rangle \mid [\langle OE \rangle \langle A' \rangle$
 $\langle SST_F1' \rangle \rightarrow ID \langle ID \rangle \langle SST_F1'' \rangle$
 $\langle SST_F1_obj \rangle \rightarrow ID = new \ ID(\langle condition \rangle); \mid \langle ID \rangle \langle SST_F1'' \rangle$

```

<SST_F1''>→[<OE><arr_call>] <ID_arr>| <Asg_Op_MST> | DI;
|(<condition>);
<ID_arr> → <Asg_Op_MST> | DI ;
<Asg_Op_MST> → <AsgOp_Equal> <OE> <Asg_Op_MST>|;

```

```

<non_class_MST> → <SST_C><non_class_MST> | null
<SST_C>→AM <SST_C'>|<object>
<SST_C'> → Static DT ID <list_attributes>|DT < SST_C''>| void ID
(<parameter>){<non_func_MST>}| ID (<parameter>)<base>{<MST_constructor>}
< SST_C''>-> ID<SST_C'''>|<C' > ID (<parameter>){<non_func_MST>}
< SST_C'''>-><SST_F''>|(<parameter>){<non_func_MST>}

```

```

<MST_constructor> → <MST_constructor'><MST_constructor>|null
<MST_constructor'> →DT ID <SST_F''> | this. <SST_F1'>|ID <SST_F1_obj> |
<for_loop> | <while_loop> | <if-else> | <TCF> | DI ID <ID_array>|return;

```

```

<list_attributes>→ <list_attributes1>| = <L_A'>
<list_attributes1> → ; | ,ID<list_attributes>
<L_A'>→<OE_class_MST> <list_attributes>

```

```

<OE_class_MST>→<A_OE_class_MST ><OE_class_MST'>
<OE_class_MST'>→OR<A_OE_class_MST ><OE_class_MST'>|€
<A_OE_class_MST >→<R_OE_class_MST ><A_OE_class_MST'>
<A_OE_class_MST'>→AND <R_OE_class_MST ><A_OE_class_MST'>|€
<R_OE_class_MST >→<E_OE_class_MST ><R_OE_class_MST'>
<R_OE_class_MST'>→ROP<E_OE_class_MST ><R_OE_class_MST'>|€
<E_OE_class_MST >→<T_OE_class_MST ><E_OE_class_MST'>
<E_OE_class_MST'>→PM<T_OE_class_MST ><E_OE_class_MST'>|€
<T_OE_class_MST >→<F_OE_class_MST ><T_OE_class_MST'>
<T_OE_class_MST'>→MDM<F_OE_class_MST ><T_OE_class_MST'>|€
<F_OE_class_MST >→<const>|!(<OE_class_MST >)

```