**1) Syntax rules**

The following BNF describes the Rat17F.

<Rat17F> ::= <Opt Function Definitions>

%% <Opt Declaration List> <Statement List>

<Opt Function Definitions> ::= <Function Definitions> | <Empty>

<Function Definitions> ::= <Function> | <Function> <Function Definitions>

<Function> ::= @ <Identifier> (<Opt Parameter List> ) <Opt Declaration List> <Body>

<Opt Parameter List> ::= <Parameter List> | <Empty>

<Parameter List> ::= <Parameter> | <Parameter> , <Parameter List>

<Parameter> ::= <IDs > : <Qualifier>

<Qualifier> ::= integer | boolean | floating

<Body> ::= { < Statement List> }

<Opt Declaration List> ::= <Declaration List> | <Empty>

<Declaration List> := <Declaration> ; | <Declaration> ; <Declaration List>

<Declaration> ::= <Qualifier > <IDs>

<IDs> ::= <Identifier> | <Identifier>, <IDs>

<Statement List> ::= <Statement> | <Statement> <Statement List>

<Statement> ::= <Compound> | <Assign> | <If> | <Return> | <Write> | <Read> | <While>

<Compound> ::= { <Statement List> }

<Assign> ::= <Identifier> := <Expression> ;

<If> ::= if ( <Condition> ) <Statement> fi |

if ( <Condition> ) <Statement> else <Statement> fi

<Return> ::= return ; | return <Expression> ;

<Write> ::= write ( <Expression>);

<Read> ::= read ( <IDs> );

<While> ::= while ( <Condition> ) <Statement>

<Condition> ::= <Expression> <Relop> <Expression>

<Relop> ::= = | /= | > | < | => | <=

<Expression> ::= <Expression> + <Term> | <Expression> - <Term> | <Term>

<Term> ::= <Term> \* <Factor> | <Term> / <Factor> | <Factor>

<Factor> ::= - <Primary> | <Primary>

<Primary> ::= <Identifier> | <Integer> | <Identifier> [<IDs>] |

( <Expression> ) | <Real> | true | false

<Empty> ::= e

**2)Here is how this code implement:**

* So we must have all rules of grammar as methods and array that contain all token type of token and content of token.
* For example if we have temp:=4;
* So in array of content we have temp,:= ,4,;
* And array of types we have identifier, operator, integer, operator.
* All content array contain all tokens from 0 to tokens size.
* All types array contain all types of tokens .
* For example
* **<Rat17F>  ::=  <Opt Function Definitions>**
* **%%  <Opt Declaration List> <Statement List>**
* In code
* void All operation::start\_point()
* {
* Opt\_Function\_Definition();//prefix
* if(allcontent[update\_index]=="%%")//second on rule
* {
* Opt\_Declaration\_List();//third on rule
* Statement\_List();//fourth on rule
* }
* }
* **<Opt Function Definitions> ::= <Function Definitions> | <Empty>**
* **<Function Definitions>  ::= <Function> | <Function> <Function Definitions>**
* **<Function> ::=  @  <Identifier>  (<Opt Parameter List> )   <Opt Declaration List>  <Body>**
* So on opt function definitions we have two choice first is empty and second is function definition and function definition must have at least on function that begin with @ so we look here for @
* In code
* void  Alloperation::Opt\_Function\_Definition()
* {
* if(allcontent[update\_index]=="@")
* {
* myfunction();
* }
* }
* void  Alloperation::myfunction()
* {
* check\_length();//  hold @ then increase index to reach to next token
* if(alltypes[update\_index]=="identifier")
* {
* check\_length();
* if(allcontent[update\_index]=="(")
* {
* check\_length();
* Opt\_Parameter\_List();
* if(allcontent[update\_index]==")")
* {
* check\_length();
* Opt\_Declaration\_List();
* Body();
* }
* }
* }
* }