IFJ2020 Grammar rules *– by Poli*

## Terminals

ID, K\_WORD, PLUS, MINUS, TIMES, DIV, EQ, L\_EQ, G\_EQ, LESS, GREAT, N\_EQ, L\_VINC(, R\_VINCL), LEFT\_BRAC{, RIGHT\_BRAC}, IF, ELSE, FOR, IS, ASSIGN, STRING\_LITERAL, INT\_LITERAL, COMMA, DOUBLE\_LITERAL, RETURN, PACKAGE\_MAIN, Q\_MARK (uvozovky), F\_MAIN (povinná funkce main), TRUE, FALSE, EOF, EOL, FUNC(jen speciální K word..), ASSIGN, IF, ELSE, FOR, SEMICOLON

## Non-Terminals

<prog>, <func>, <param>, <param\_n>, <retval>, <retvals>, <body>, <f\_retval>, <f\_retvals>, <expr>, <func\_call>, <call-params>, <if>, <for>, <bool\_expr>, <for\_init>

## Rules

-------------------Funkce, tělo programu, definice funkcí a návratových paramatrů-------------------

* 1. <prog> *// jelikož v IFJ20 nejsou globalní promenne tak povolujeme na zacatku jen definice fci*
     1. <prog> -> PACKAGE\_MAIN <func>
     2. <prog> -> EOF
     3. <prog> -> EOL
  2. <func> *// definice funkce: func id (params) (retvals){ body } může následovat další funkce* 
     1. <func>-> FUNC ID L\_VINCL <param\_n> R\_VINCL <retvals> L\_BRAC <body> <f\_retvals> R\_BRAC EOL<func>
     2. <func> -> ε
     3. <func> ->EOL<func>
  3. <param> *// buď nejsou žádné parametry: () nebo (param)*
     1. <param> -> ε
     2. <param> -> [INT, DOUBLE, STRING] ID <param>
     3. <param>->COMMA [INT, DOUBLE, STRING] ID <param>
  4. <retvals> *// návratové hodnoty nemusí být přítomny a nebo pokračuj na retval*
     1. <retvals> -> L\_VINCL [INT, DOUBLE, STRING]<retval>R\_VINCL
     2. <retvals> -> ε
  5. <retval> *// buď jedna návratová hodnota nebo více*
     1. <retval>-> COMMA [INT, DOUBLE, STRING] <retval>
     2. <retval>-> ε
  6. <f\_retvals> *// návratové hodnoty uvnitř těla funkce, nemusí být nebo pokračuj na f\_retval*
     1. <f\_retvals> -> ε
     2. <f\_retvals> -> RETURN <expr> <f\_retval>
  7. <f\_retval> *// návratové hodnoty mohou být jeden či více výrazů*
     1. <f\_retval> -> COMMA<expr> <f\_retval>
     2. <f\_retval> -> ε

-------------------------------------------------------------------------------------------------------------------------------------

-------------------------------------------Tělo funkce, cykly, aritemtické výrazy------------------------------------------

* 1. <body> *// jsme v tělu funkce, buď přiřazujeme do proměnné výraz nebo volání funkce, také se* 
     1. <body> -> EOL
     2. <body> -> ε
     3. <body> -> <exp> EOL <body>
     4. <body> -> ID <body\_id> EOL <body>
     5. <body> -> <if> EOL<body>
     6. <body> -> <for> EOL<body>
  2. <body\_id>
     1. <body\_id> -> = <body\_id\_var>
     2. <body\_id> -> := <body\_id\_var>
  3. <body\_id\_var>
     1. <body\_id\_var> -> <expr>
     2. <body\_id\_var> -> <func-call>
  4. <func-call> *// přiřazujeme z volání funkce*
     1. <func-call> -> ID L\_VINCL <call\_params> R\_VINCL
  5. <call-params> *// nemusí být žádné paramentry z volání funkce, jinak pokračuj na call-param*
     1. <call-params> -> ε
     2. <call-params>-> <call-param>
  6. <call-param>*// buď máme jeden parametr nebo více*
     1. <call-param> -> <expr>
     2. <call-param> -> <expr> COMMA <call-param>
  7. <if>*// if obsahuje bool podmínku zároky tělo, else další závorky a tělo*
     1. <if> -> IF <bool\_expr> R\_BRAC EOL <body> L\_BRAC ELSE R\_BRAC EOL <body> L\_BRAC
  8. <for> *// for (init; condition; after)*
     1. <for> -> for L\_VINCL<for\_init> SEMICOLON <bool\_expr> SEMICOLON <expr> R\_VINCL L\_BRAC EOL <body> R\_BRAC
  9. <for\_init> *// to co může být v první podmínce ve for*
     1. <for\_init> -> [INT, DOUBLE, STRING] ID ASSIGN <expr>