

```
PROGRAM ::= [VARSEQ] main { INSTRSEQ }
VARSEQ ::= ( VARDECL ; ) *
VARDECL ::= var IDLIST : ( integer | float | boolean | string )
IDLIST ::= IDENT ( , IDENT ) *
EXPR ::= [ + | - ] SIMPEXPR ( RELOP SIMPEXPR ) *
SIMPEXPR ::= TERM ( ADDOP TERM ) *
TERM ::= FACTOR ( MULTOP FACTOR ) *
FACTOR ::= IDENT | LSTRING | LINT | LBOOL | LREAL
RELOP ::= == | < | > | != | >= | <=
ADDOP ::= + | - | or
MULTOP ::= * | / | and
INSTRSEQ ::= ( INSTR ; ) *
INSTR ::= ASSIGN | COND | LOOP | IN | OUT | BLOCK
ASSIGN ::= IDENT = EXPR
OUT ::= print ( EXPR )
IN ::= read ( IDENT )
COND ::= if ( EXPR ) { INSTRSEQ }
LOOP ::= do ( EXPR ) { INSTRSEQ }
BLOCK ::= block { [ VARSEQ ] INSTRSEQ }
```

Oznaczenia:

- Czcionką pogrubioną zaznaczono symbole terminalne (proszę zwrócić uwagę na pojedyncze znaki, niektóre też są terminalami)
- [] oznacza "opcjonalnie"
- () * oznacza "zero lub więcej razy"
- | oznacza "lub" (czyli kilka produkcji w jednym wierszu)
- IDENT - identyfikator, LSTRING - literał napisowy, LINT - literał całkowity itd.

Uwagi:

- Najprostsza gramatyka zawierająca instrukcje: podstawienia, warunkową, pętli, wejścia/wyjścia oraz bloki z lokalnymi danymi. Dla konstrukcji "blok" zakładamy, że jego zmienne "przykrywają" zmienne globalne.
- Dodatkowe założenia leksykalne, składniowe, semantyczne mogą być określone przez studenta **po uprzedniej konsultacji z prowadzącym pracownię**.
- Generacja kodu wynikowego - do wyboru: pseudokod lub kod maszynowy wybranego procesora. Pseudokod wymaga implementacji maszyny wirtualnej
- Gramatyka może wymagać przekształceń lub uzupełnień - wszelkie wątpliwości należy wyjaśniać z prowadzącymi