JPP - Interpreter

Opis języka

Karol Soczewica (ks394468)

Opis

Język, którego interpreter będę implementował to język Latte z pewnymi dodatkami/zmianami (Latte++). Wykonywanie programu będzie zaczynało się od funkcji **int main() {}**, która w programie musi wystąpić.
W Latte++ dodatkowym typem będzie jednowymiarowa tablica.
Dodatkami do instrukcji Latte są instrukcje **for**, **foreach** (tak jak w Javie), **break**, **continue** oraz **print**. Oprócz tego język Latte++ będzie miał słowo kluczowe **final**, dzięki któremu będzie można mieć zmienne tylko do odczytu, a intrukcja **if** zostanie rozszerzona o możliwość dodawania bloków **else if**.
Zmianą w składni w stosunku do Latte jest to, że instrukcje w pętlach oraz w **if** muszą zawsze znajdować się w środku bloku, czyli nielegalne będzie napisanie **if (i < 5) print(i)**, zamiast tego trzeba będzie napisać **if (i < 5) { print(i) }**.
Każda zmienna musi być zadeklarowana przed użyciem. Jeśli zmienna nie jest zainicjalizowana, to przyjmuje domyślną dla swojego typu wartość: **int** → 0, **bool** → false, **string** → "".
Zmienne deklarowane wewnątrz bloków przesłaniają zmienne o tych samych nazwach spoza bloku.
Parametry funkcji przekazywane są przez wartość.

Gramatyka

– Programs ————————————

entrypoints Program ;
Program. Program ::= [FunDef] ;

– Types ————————————

Int. Type ::= "int" ;
Str. Type ::= "string" ;
Bool. Type ::= "bool" ;
Void. Type ::= "void" ;
Array. Type ::= "Array" "<" Type ">" ;

– Expressions ————————————

Evar. Expr6 ::= Ident ;
ELitInt. Expr6 ::= Integer ;
ELitTrue. Expr6 ::= "true" ;
ELitFalse. Expr6 ::= "false" ;
EApp. Expr6 ::= Ident "(" [Expr] ")" ;
EString. Expr6 ::= String ;
Neg. Expr5 ::= "-" Expr6 ;
Not. Expr5 ::= "!" Expr6 ;
EMul. Expr4 ::= Expr4 MulOp Expr5 ;
EAdd. Expr3 ::= Expr3 AddOp Expr4 ;
ERel. Expr2 ::= Expr2 RelOp Expr3 ;

```
EAnd. Expr1 ::= Expr1 "&&" Expr2 ;
EOr. Expr ::= Expr "||" Expr1 ;
coercions Expr 6 ;
separator Expr "," ;

-- Operators ----------------------------

Plus. AddOp ::= "+" ;
Minus. AddOp ::= "-" ;
Times. MulOp ::= "*" ;
Div. MulOp ::= "/" ;
Mod. MulOp ::= "%" ;
Lt. RelOp ::= "<" ;
Leq. RelOp ::= "<=" ;
Gt. RelOp ::= ">" ;
Geq. RelOp ::= ">=" ;
Eq. RelOp ::= "==" ;
Neq. RelOp ::= "!=" ;

-- Comments ----------------------------

comment "#" ;
comment "//" ;
comment "/*" "*/" ;
```