

SEMANTYKA I WERYFIKACJA - Zadanie egzaminacyjne nr 1

Napisz semantykę operacyjną dużych kroków języka o gramatyce:

$$\begin{aligned} Num \ni n &::= 0 \mid 1 \mid -1 \mid 2 \mid -2 \mid \dots \\ Var \ni x &::= x \mid y \mid \dots \\ PName \ni p &::= p \mid q \mid \dots \\ Expr \ni e &::= n \mid x \mid e_1 + e_2 \mid e_1 * e_2 \mid e_1 - e_2 \\ BExpr \ni b &::= \text{true} \mid \text{false} \mid e_1 < e_2 \mid e_1 = e_2 \mid b_1 \wedge b_2 \mid \text{not } b \\ Decl \ni d &::= \text{var } x = e \mid \text{proc } p \mid d_1; d_2 \\ Instr \ni I &::= x := e \mid I_1; I_2 \mid \text{if } b \text{ then } I \mid \text{while } b \text{ do } I \mid \text{begin } d; I \text{ end} \mid \text{call } p(I) \mid \text{doit} \end{aligned}$$

Jest to język z rekurencyjnymi procedurami, przyjmującymi instrukcje jako (nienazwane) argumenty. Instrukcja `call p(I)` wywołuje procedurę `p`, przekazując jej instrukcję `I` jako argument. Instrukcja `doit`, wykonana w ciele procedury, powoduje wykonanie instrukcji przekazanej jako argument do tej procedury. Odbywa się to zgodnie z zasadami statycznej widoczności identyfikatorów. Na przykład na końcu wykonania bloku

```
begin
  var x=0; var y=0;
  proc p
    x := x+1; doit;
  begin
    var x=0;
    call p(x := x+2);
    y := x
  end
end
```

zmienna `x` zadeklarowana w drugim wierszu przyjmuje wartość 1, a zmienna `y` – wartość 2. Sama instrukcja `doit` także jest interpretowana statycznie, tzn. oznacza parametr tej procedury, w której ta instrukcja występuje jako fragment tekstu. Na przykład na końcu wykonania wykonaniu bloku

```
begin
  var x=0;
  proc p
    x := x+1; doit;
  proc q
    call p(x := x+2; doit);
  call q(x := x+3);
end
```

zmienna `x` przyjmuje wartość 6.

Semantyka pozostałych konstrukcji jest standardowa.

W rozwiązaniu należy podać co najmniej reguły wyprowadzania dla instrukcji `call` i `doit`, a także dla deklaracji procedur. Pozostałe reguły można pominąć jeżeli są standardowe lub zupełnie analogiczne do standardowych. Należy natomiast podać definicje wszystkich dziedzin pomocniczych i ogólną postać konfiguracji i relacji ewaluacji dla każdej kategorii syntaktycznej.

Można używać semantycznej funkcji *alloc*, która dla danego stanu pamięci zwraca pewną nieużywaną w tym stanie komórkę pamięci. Można też założyć, że programy nie korzystają z niezadeklarowanych zmiennych ani procedur.