Programski jezik PINS'26

1 Leksikalna pravila

Programski jezik PINS'26 je pisan v abecedi ASCII in vsebuje naslednje leksikalne enote:

- *Konstante*:
  - *število*:
    Neprazno zaporedje desetiških števk, pred katerimi lahko stoji predznak (`+` or `-`).
  - *znak*:
    Znak zapisan v enojnih navednicah (`'`). Znak je lahko (a) katerikoli ASCII znak s kodo v obsegu $\{32...126\}$, pri čemer morata biti enojna navednica in obratna poševnica (\) uvedeni z obratno poševnico, (b) znak za konec vrstice, ki je zapisan kot `\n`, ali (c) katerikoli ASCII znak zapisan s kodo v obliki `\xx`, pri čemer je `X` šestnajstiška števka (`0`...`9` in `a`...`f`).
  - *niz znakov*:
    Niz znakov, lahko prazen, zapisan v dvojnih navednicah (`"`). Znak je lahko (a) katerikoli ASCII znak s kodo v obsegu $\{32...126\}$, pri čemer morata biti dvojna navednica in obratna poševnica (\) uvedeni z obratno poševnico, (b) znak za konec vrstice, ki je zapisan kot `\n`, ali (c) katerikoli ASCII znak zapisan s kodo v obliki `\xx`, pri čemer je `X` šestnajstiška števka (`0`...`9` in `a`...`f`).
- *Simboli*:
  `=` `;` `,` `&&` `||` `!` `==` `!=` `>` `<` `>=` `<=` `+` `-` `*` `/` `%` `^` `(` `)`
- *Imena*:
  Neprazno zaporedje črk (`A`...`Z` in `a`...`z`), desetiških števk (`0`...`9`) in podčrtajev (`_`), ki (a) se začne s črko ali podčrtajem in (b) ni ključna beseda.
- *Ključne besede*:
  `fun` `var` `if` `then` `else` `while` `do` `let` `in` `end`
- *Komentarji*:
  Niz znakov, ki se začne z `//` in se konča na koncu vrstice.
- *Belo besedilo*:
  Presledek in znaki HT, LF in CR. Konec vrstice označuje znak LF, HT je širok 8 znakov.

Leksikalni elementi morajo biti razpoznani od leve proti desni po pravilu najdaljšega ujemanja na skrajno levem mestu.

2 Sintaksna pravila

Sintakso programskega jezika PINS'26 določa kontekstno neodvisna gramatika z začetnim simbolom *program* in naslednjimi produkcijami:

program
$\longrightarrow$ (*definition*)*

definition
- ⟶ `fun` IDENTIFIER `(` *parameters* `)`
- ⟶ `fun` IDENTIFIER `(` *parameters* `)` `=` *statements*
- ⟶ `var` IDENTIFIER `=` *initializers*

parameters
- ⟶ (IDENTIFIER (`,` IDENTIFIER)*)?

statements
- ⟶ *statement* `;` (*statement* `;`)*

statement
- ⟶ *expression*
- ⟶ *expression* `=` *expression*
- ⟶ `if` *expression* `then` *statements* (`else` *statements*)? `end`
- ⟶ `while` *expression* `do` *statements* `end`
- ⟶ `let` (*definition*)+ `in` *statements* `end`

expression
- ⟶ INTCONST | CHARCONST | STRINGCONST
- ⟶ IDENTIFIER (`(` *arguments* `)`)?
- ⟶ prefix-operator *expression*
- ⟶ *expression* pstfix-operator
- ⟶ *expression* binary-operator *expression*
- ⟶ `(` *expression* `)`

arguments
- ⟶ (*expression* (`,` *expression*)*)?

initializers
- ⟶ (*initalizer* (`,` *initializer*)*)?

initializer
- ⟶ (INTCONST `*`)? *const*

const
- ⟶ INTCONST | CHARCONST | STRINGCONST

Simboli INTCONST, CHARCONST in STRINGCONST označujejo celoštevilske konstante, znakovne konstante in nize, zaporedoma.

Prioriteta operatorjev je sledeča

postfiksni operatorji	`^`	NAJVIŠJA PRIORITETA
prefiksni operatorji	`! + - ^`	
multiplikativni operatorji	`* / %`	
aditivni operatorji	`+ -`	
primerjalni operatorji	`== != < > <= >=`	
konjunkcija	`&&`	
disjunkcija	`||`	NAJNIŽJA PRIORITETA

Multiplikatini in aditivni operatorji so levo asociativni,primerjalni operatorji niso asociativni.

3 Semantična pravila

Imena. Vsa imena so v istem imenskem prostoru.

Vsako ime je vidno v celotnem območju dosega, v katerem je definirano. Območje dosega je ustvarjeno na tri načine:

1. Vse definicije imen zunaj funkcij so v globalnem območju dosega.
2. Funkcija ustvari novo območje dosega (ime funkcije je zunaj ustvarjenega dosega).
3. Stavek `let-in` ustvari novo območje dosega (vse lokalne definicije so znotraj območja obsega).

Tipi. Obstaja en sam podatkovni tip: 32-bitno predznačeno število v dvojiškem komplementu, ki služi kot vrednost in kot naslov. Znakovne konstante predstavljajo cela števila glede na ASCII kodiranje. Nizi znakov predstavljajo naslov, na katerem so shranjena števila, ki so dobljena iz znakov niza glede na ASCII kodiranje.

Zaporedje stavkov, ki tvori telo funkcije, se mora končati

1. s stavkom, ki vsebuje zgolj izraz, ali
2. z `let-in` stavkom, katerega zaporedje stavkov se konča s stavkom pod točko 1 ali 2.

Leve vrednosti. Leva vrednost je

1. spremenljivka ali
2. izraz, katerega najbolj zunanji operator je postfiksni `^`.