LFTC

Friday, 6 October 2017 16:53

LISP like language specification:

() are part of the language. They wrap every statement or expressio

```
<program> ::= <decls><stmts>

<decls> ::= <decl>|<decl><decls>
    <decl> ::= <declv>|<decl type>
        <declv> ::= (newv <id> <type> <expr>) | (newv <id> <type>)
            <id> ::= [a...z]+[0-9]*
            <type> ::= integer|real
        <decl type> ::= (struct <id> <decls>)

<stmts> ::= <stmt>|<stmt><stmts>
    <stmt> ::= <assig>|<while>|<if>|<read>|<write>
        <assig> ::= (= <id> <expr>|<const>)
            <const> ::= <integer>|<real>
            <expr> ::= (<op> <params>)
                <op> ::= + | - | * | / | % | ** | < | > | <= | >=
                <params> ::= <param>|<param><params>
                <param> ::= <id>|<const>|<expr>
        <while> ::= (while <expr> <stmt>|(do <stmts>))
        <if> ::= (if <expr> <stmt>|(do <stmts>) <stmt>|(do <stmts>
        <read> ::= (read <id>) | (read)
        <write> ::= (write <id>) | (write <const>) | (write <expr>
```

Prog1:

```
    (newv radius real)
    (read radius)
    (write (* 3.14 radius radius))
    (write (* 2 3.14 radius))
```

Prog2:

```
    (newv a integer 9)
    (newv b integer 12)
    (newv r integer)
    (while (<> b 0)
        (do
            (= r b)
```

n.

| == | <>

))

)

```
            (= b (% a b))
            (= a r)))
    (write a)
```

Prog3:

```
    (newv sum real 0)
    (newv n integer)
    (newv i integer 0)
    (read n)
    (while (< i n)
        (do
            (= sum (+ sum (read)))
            (= i (+ i 1))))
    (write sum)
```