

Semestrální práce

-

KIV/FJP

Zadání: 1. Tvorba překladače zvoleného jazyka

Jazyk: Podmnožina jazyka ANSI C

Architektura: Instrukce PL/0

Tým: P. Patera, L. Kaňák

Gramatika

- Datové typy

```
data_type
:   'string'
|   'int'
|   'boolean'
;
```

- String – řetězec znaků

```
str
:   '"' (~SPECIAL_CHARS | ESCAPE)* '"'
;
```

- **Int** – celé číslo

```
int
  : NUMERIC+
  ;
```

- **Real** – reálné číslo

```
real
  : int ( '.' int ) ?
  ;
```

- **Boolean** – logický datový typ

```
boolean
  : 'true' | 'false' | ID
  ;
```

- **Array** – pole

```
array
  : data_type '[' ']'
  ;
```

- Proměnná

```
var_type
  : data_type
  | array
  ;
```

- Definice a deklarace

```
def
  : ID '=' (ternar_oper | value)
  ;
```

```
multiple_def
  : ID ('=' ID)* '=' (value | ternar_oper)
  ;
```

```
declar
  : var_type (ID | def | multiple_def) ';'
  ;
```

```
const_declar
  : 'const' var_type (def | multiple_def)
  ;
```

```
value
    : (ID | num_def | str_def | boolean_def
        | array_def)
    ;

num_def
    : sign? term (sign term)*
    ;

str_def
    : str ('+' str)*
    ;

boolean_def
    : factor (bin_oper factor)*
    ;

array_def
    : '{' (value (',' value)*)? '}'
    ;

sign
    : '-' | '+'
    ;
```

- Výrazy

```
term
: factor (('*' | '/' | '&' | '|') factor)*
;
```

```
factor
: (num | boolean)
| '!'? '(' (num_def | boolean_def) ')'
| '!' (num | boolean)
;
```

```
num
: int
| real
| ID
;
```

- Operátor

```
comp_oper
: '==' | '>=' | '<=' | '>' | '<' | '!=' | '==='
;
```

```
bin_oper
: '&&' | '||'
;
```

```
ternar_oper
: cond_head '?' value ':' value
;
```

- Podmínka

```
cond
: 'if' '(' cond_head ')' '{' block '}'
  ( 'else' '{' block '}' )?
;
```

```
cond_head
: ((num_def comp_oper num_def ) | boolean_def)
  (( bin_oper num_def comp_oper num_def) |
   boolean_def)*
;
```

- **Cykly**

```
loop
: loop_while | loop_for | do_while | foreach
;
```

- **While**

```
loop_while
: 'while' '(' cond_head ')' '{' block '}'
;
```

- **Do while**

```
do_while
: 'do' '{' block '}' 'while' '(' cond_head ')' ';'
;
```

- **For**

```
loop_for
: 'for' '(' data_type def ';' cond_head ';'
def ')' '{' block '}'
;
```


- **Foreach**

```
foreach
    : 'foreach' '(' var_type ID ':' ID ')' '{'
    block '}'
    ;
```

- **Switch**

```
s_switch
    : 'switch' '(' ID ')' '{'
    (('case' (num_def | str_def) ':' + block 'break'?) +
    |
    (('default:' + block 'break'?)? '}'
    ;
```

- **Funkce**

```
func_def
    : (var_type | 'void') ID '(' (param | 'void') ')'
    '{' block '}'
    ;
```

- Parametry funkce

```
param
    : (var_type ID (',' var_type ID)*)?
    ;
```

- Volání funkce

```
call_fnc
    : func ';'
    ;
```

- Návrat z funkce

```
r_return
    : 'return'
      ( ternar_oper
        | value
        | func
        )
      ';'
    ;
```

- Blok

Block

```
: ( declar
  | const_declar
  | def
  | loop
  | cond
  | s_switch
  | call_fnc
  ) *
r_return?
;
```

ID
: (ALPHABET | CHARS) (ALPHABET_NUMERIC | CHARS) *
;

NUMERIC
: ('0' .. '9')
;

CHARS
: ' _'
;

ALPHABET
: ('A' .. 'Z' | 'a' .. 'z')
;

ALPHABET_NUMERIC
: ALPHABET | NUMERIC
;

SPECIAL_CHARS
: ' " '
;

ESCAPE
: '/' SPECIAL_CHARS
;

Děkujeme za pozornost