UNIVERSIDADE ESTADUAL DO OESTE DO PARANÁ
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
COLEGIADO DE CIÊNCIA DA COMPUTAÇÃO

ESPECIFICAÇÃO DA LINGUAGEM C MAIS OU MENOS E ANALISADOR LÉXICO

PEDRO XAVIER, VALQUÍRIA BELUSSO

CASCAVEL - 2022

SUMÁRIO

Especificação da Linguagem: C Mais ou Menos

1. Tipos de dados suportados

- Inteiros:
    - Inteiro com sinal (complemento de dois)
        - i8, i16, i32, i64.
    - Inteiro sem sinal
        - ui8, ui16, ui32, ui64.
- Ponto Flutuante:
    - f32, f64.
- Booleanos:
    - bool: byte único, false se zero, true caso contrário.

2. Operadores suportados:

2.1 Atribuição

É realizada com o símbolo =, podendo ser feita das seguintes maneiras:

- <tipo> <id> = <valor>, <id> = <valor>;
- <tipo> <id> = <valor>, <id>;
- <tipo> <id>, <id> = <valor>;
- <tipo> <id>, <id>;

2.2 Atribuição aritmética

É possível atualizar o valor de uma variável usando a seguinte expressão:

- <variável> <operador> <valor>

Atribuição	Operador
soma	+=
subtração	-=
multiplicação	*=
divisão	/=
módulo	%=

2.3 Operadores aritméticos

Operação	Operador	Sintaxe
soma	+	i + j
subtração	-	i - j
multiplicação	*	i * j
divisão	/	i / j

módulo	%	i % j

2.4 Operadores Lógicos

As operações lógicas resultam em um valor booleano. Para operandos não booleanos, zero significa falso, caso contrário, verdadeiro.

Operação	Operador	Sintaxe
E	&&	i && j
Ou	| |	i | | j
Não	!	! i

2.5 Operadores Relacionais

Expressões de comparação resultam em um valor booleano.

Operação	Operador	Sintaxe
Igual	==	i == j
Diferente	!=	i != j
Maior ou igual	>=	i >= j
Maior	>	i > j
Menor ou igual	<=	i <= j
Menor	<	i < j

3. Incremento e Decremento

Operação	Operador	Sintaxe
Incremento	++	i ++
Decremento	--	i --

4. Literais

Há quatro formas de literais utilizados:

- String Literal: Qualquer texto entre "" é um literal de string. Não pode ser atribuído a uma variável, mas pode ser usado com a instrução write. Exemplo: "Olá!".
- Integer Literal: Qualquer sequência de números, uso de outras bases. Exemplo: 42, -13, 0xdeadbeef, 0o777, 0b1010 ...

- Float Literal: Qualquer sequência de números começando com um ponto ou com um ponto na sequência. Exemplo: 4.2, .5, -2.0. -.5 …
- Booleano Literal: true, false.

5. Comentários

Existe uma forma de escrever algo no programa de forma que este ignore o que foi escrito, essa forma é através dos comentários utilizando o símbolo #, sendo o compatível com expressões shebang em arquivos de script.

6. Estruturas suportadas pela linguagem

6.1 Estruturas de decisão

A linguagem utiliza as seguintes estruturas de decisão:

- if: verifica uma declaração (entre parênteses) e se for verdadeira, executa o seguinte bloco de código (entre chaves);
- else: deve estar após uma instrução if e suas instruções/bloco de código, executa um bloco de código a seguir se a condição na instrução if for falsa;
- elif: equivalente a um if logo depois do else de outro if, deve estar depois de uma instrução if ou elif, tem sua própria expressão condicional e bloco de código, pode ser seguido por outro elif ou finalmente por um else.

Exemplo:

```
if (age < 18) {
  write "can't drink or drive"
} elif (age < 99) {
  write "can drink and drive"
} else {
  write "can't play with legos"
}
```

6.2 Estruturas de repetição

Duas estruturas de repetição são utilizadas na linguagem, sendo eles for e while. Exemplo:

```
i32 i = 0;
while (i < 2) {
  write "batata\n";
}

for (i = 0; i < 3; i += 1) {
  write "batata\n"
}
```

7. Identificadores e palavras-chave

7.1 Identificadores

As variáveis devem ser identificadas com nomes exclusivos, esses nomes exclusivos são chamados de identificadores. Um identificador deve começar com uma letra ou sublinhado e pode seguir com qualquer sequência de caracteres alfanuméricos (mais sublinhado).

Exemplos: i32 minutes = 60;
i32 _something = -1;
i32 MyVariableIs = 0;
i32 _123 = 123;

7.2 Declarações de ES

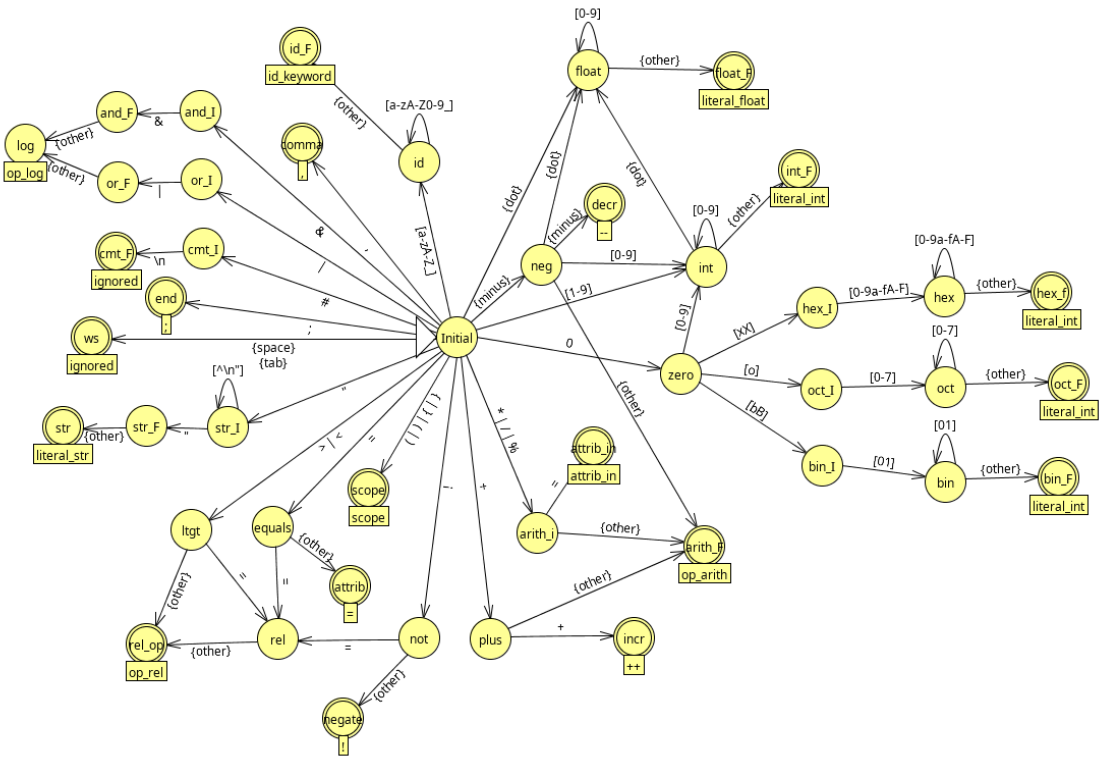
- read: lê de stdin e define o valor na variável. Sintaxe: read <variável>;
- write: escreve em stdout, pode escrever variáveis e literais, também é possível escrever várias expressões separadas por vírgulas. Sintaxe write <variáveis>;

8. Palavras reservadas

São palavras reservadas:

- tipos: i8, ui8, i16, ui16, i32, ui32, i64, ui64, f32, f64, bool.
- booleanos literais: true e false.
- estruturas de fluxo: if, else, elif, for, while.
- comandos de ES: read e write.

9. Automatão



10. Exemplo de código

```
# fatorial

i32 n = 5, i = 1;

write "calculando fatorial de ", n, "\n";

while ( n > 1) {
    i *= n;
    n--;
}

write "resultado: ", i, "\n";
```

```
# fatorial

i32 n, res = 1;
write "entre com um numero\n";
read n;

if ( n > 0 ) {
    write "calculando fatorial de ", n, "\n";
    for (i32 i = 2; i <= n; i++) {
        res *= i;
    }
    write "resultado: ", res, "\n";
} else {
    write "entrada invalida\n";
}
```

11. BNF

<program>::= <stmts>
<stmts>::=
| <stmt>
| <stmt> <stmts>
| ε
<stmt>::=
| <decl>
| <attrib>
| <expr>
| <command>
<decl>::=
| <type> <id>;
| <type> <id> = <expr>;
<attrib>::=
| <var> = <expr>;
| <var> <attrib_in> <expr>;
<expr>::=
| <arith_expr>
| <expr_bool>
<arith_expr>::=
| <arith_expr> + <arith_term>
| <arith_expr> - <arith_term>
| <arith_term>
<arith_term>::=
| <arith_term> * <arith_term>
| <arith_term> / <arith_term>
| <arith_term> % <arith_term>
| (<arith_expr>)
| <var>
| <literal>
<expr_bool>::=
| ! <expr_bool>
| <var> <rel_op> <var>
| (<expr_bool>) <op_bool> (<expr_bool>)
| <boolean> <op_bool> <boolean>
| <boolean>
<boolean>::=
| <literal_bool>
| <var>
<command>::=
| read <var> ;
| write <writables> ;
| <if>
| <while>
| <for>

<writables>::=

| <writable>

| <writable> , <writables>

<writable>::=

| <literal>

| <var>

| <expr>

<literal>::=

| <literal_int>

| <literal_float>

| <literal_bool>