Linguagem criada em JavaCC

Vinicius Gazolla Boneto, Icaro Peretti

O que foi feito ?

- [x] Comentário (//)
- [x] Declaração de variável
- [x] Comandos de repetição (WHILE)
- [x] Comandos de decisão (IF,ELSEIF,ELSE)
- [x] Função
- [ ] Saída de dados

Extras

- [x] Switch
- [x] Do While
- [ ] If inline /decisão ternária
- [ ] Comando inovador complexo

Entradas Valídas

O que faz ?	Entrada
Ignora algumas entradas	" ", "\t", "\n", "\t"
Abre Comentários	//
Encerra linha de argumento	;
Realizar incremento e decremento	++, --
Função	function
Comando de decisão	if
Comando de decisão encadeado	else, elseif
Repetição	while, do while
Números	[0-9], [0-9]
Letras	[a-z], [A-Z]
Operador lógico	>.<, ==, !=
Operador aritimético	+, -, *, /
Abertura e fechamento de comando	{, }
Abertura e fechamento parenteses	(,)
Atribuição	=

Tipos

Identificador	Valor
boolean	true, false
int	Números Inteiros
float, double	Números Decimais
String	Texto
byte	byte
char	Caracter

Instruções

> ### Comentários
>
> Os comentários devem ser iniciados por duas barras //.

Exemplo

```
// Este é um comentário
```

Declarar variável e atribuição de valor

As variáveis são declaradas com identificador de seu tipo e/ou com seu respectivo valor.

Exemplo

```
int a = 10;
String texto = "Ola mundo";
int b, c, d, e = 0;
```

Expressões aritimética

As expressões aritiméticas são realizas através dos operadores aritiméticos e/ou dos operadores de incremento e decremento.

Exemplo

```
int a = 20;
int b;
b = 30 + a;
b++;
```

Comandos de decisões

Os comandos de decisões são realizados através do `switch case` ou `if`, este pode ser encadeado com o `elseif` e/ou `else`. Com exessão do `else`, os outros comandos devem ser seguidos com a condição (operação lógica) declarada entre parentêses e todos devem conter a abertura de chaves. No caso do `switch` deve ser estabelecido logo em seguida a variável condicional entre parênteses, e em seguinda seus casos de uso, onde cada caso (`case`) será estabeleico os comando de execução por dois pontos(`:`) e encerrado com um `break`.

Exemplo

```
// Comando if
int a = 0;
int b = 1;
if (a == 1) {
    // Comandos
} elseif (a > b) {
    // Comandos
} else {
    //Comandos
}

// Comando switch case
switch (a) {
    case 1:
        b = b + 1
        break;
```

```
    case: 2:
        b = b + 3
        break;
}
```

Comandos de repetição

Os comandos de repetição podem ser realizados através do `while` e `do while`. Seguidos de uma operação logíca declarada entre parentêses e abertura de chaves.

Exemplo

```
// Comando while
int a = 0;
while (a < 10) {
    // Comandos
    a++;
}
```

```
// Comando do while
int a = 0;
do {
    // Comandos
    a++;
} while (a < 10);
```

Funções

As funções devem ser iniciadas com a palavra `function`. Seguido com os parâmetros declarados entre parentêses e abertura de chaves.

Exemplo

```
// Este é uma função sem parâmetros
function () {
    // Comandos
}
```

```
// Esta é uma função com parâmetros
function (int value, double otherValue) {
    // Comandos
}
```

Relato Aprendizado

As Dificuldade desse projeto é principalmente determinar os tokens e utilizalos corretamente de forma que não ocorra conflitos e erros, além de claro a configuração do ambiente que não é muito simples. Mas em compensação o trabalho te proporciona uma boa ídeia de como é criado as validações lexicas de uma linguagem e te da uma base sobre como funciona expressões regulares que são constantemente utilizadas em diversos ramos da computação.