

Lógica da Computação

Criação de Linguagem

Motivação

Linguagem baseada em Assembly

A ideia da linguagem partiu da inspiração de comandos Assembly, como visto na disciplina de Arquitetura de Computadores, visando aproximar os alunos da programação em assembly, abrindo mão de parte do rigor exigido para se familiarizarem com o formato.

Para isso busquei uma implementação mais de alto nível mas ainda com a simplicidade de comandos e clareza nos verbos de cada linha de instrução

Características

Formato

Todas as linhas tem o mesmo formato, VERBO Arg1, Args2-n
Os verbos indicam loops, criações de funções e variaves e as funções padrões, como print e leitura de input

Funções

A criação de funções foi implementada de forma que exista uma variavel global RET destinada ao retorno de funções, que deve ser escrita nas funções com o valor a ser retornado e lida em qualquer instancia.

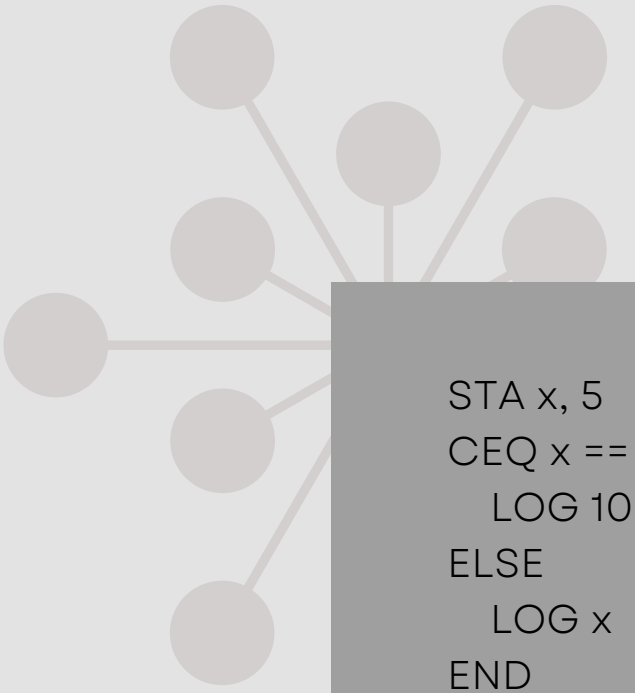
Tipagem

Trazendo elementos do assembly, a linguagem trabalha apenas com inteiros.

Vírgulas

O uso de parenteses foi implementad apenas para notação algebrica, de forma que a separação de argumentos em instruções ocorre pela orgem dor argumentos e a separação por vírgulas.

Exemplos



```
STA x, 5
CEQ x == 10
  LOG 10
ELSE
  LOG x
END
```


If/Else

Os comparativos foram implementados no formato "CEQ" + RelExp, sendo demarcados pelo token "END"

```
FUNC TESTE, A, B, C
STA RET, B+C*(2*2)
END
STA VAR, 2
CEQ VAR==0
JMP TESTE, 1, 2, 3
ELSE
JMP TESTE, 4, 5, 6
END
STA A, RET
```

Funções

A definição de uma função segue a ordem "FUNC" seguidos do nome da função e dos argumentos recebidos, e o retorno dela deve se salvo na variavel RET, como mostra o exemplo. A chamada da função ocorre a partir do token "JMP" seguido do nome e argumentos, com o retorno vai para RET, não existe atribuição nessa linha.



```
LDI SALVA
WHL SALVA <= 10
STA SALVA, SALVA+1
END
```

While

OS loops While funcionam de forma similar os If. Os inputs do usuaio são lidos pelo comando "LDI" seguido da variavel onde deve ser salvo.

