LÓGICA DA COMPUTAÇÃO, INSPER 2024.1

Linguagem de Programação

Antônio Amaral Egydio Martins

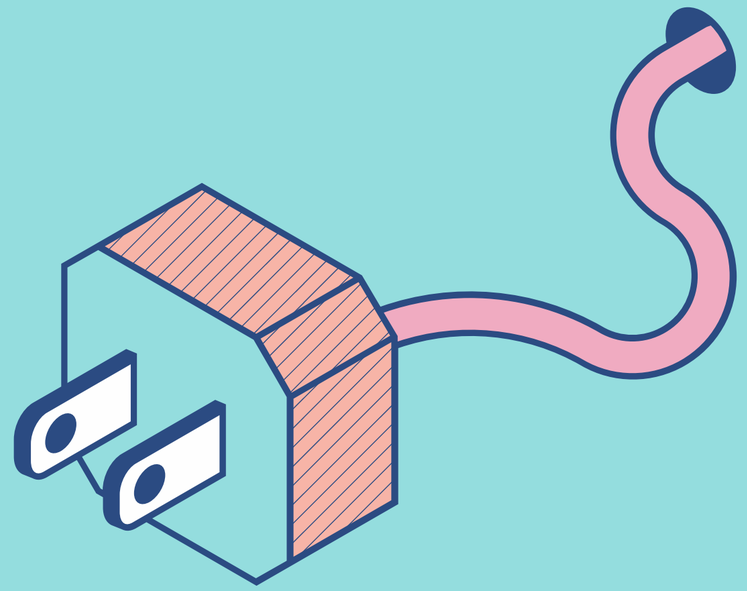
Origem da Ideia

Questão norteadora: Quais itens de meu dia a dia utilizam linguagem de programação, mesmo que de forma básica?

1. Geladeiras, televisões, computadores...

A escolhida: Controle de um ar condicionado.

Objetivo: Criar uma linguagem de programação que permita simular o funcionamento de um controle de ar condicionado.



Itens Necessários para a Linguagem

1 VARIÁVEIS

A definição de variáveis é essencial para criar linguagens de programação com armazenamento local

2 ARITMÉTICA

É necessário poder realizar operações matemáticas simples como adição, subtração, divisão e multiplicação

3 LÓGICA

Para simular um ambiente real, precisam haver comparações booleandas (Ex. AND, OR, BELOW, ABOVE e EQUAL)

4 LOOPS

É possível um ar condicionado ter modo "automático", sendo assim há necessidade de checar constantemente o estado das variáveis.

5 CLASSES

É possível representar tudo isso, apenas com variáveis, porém classes aumentam a organização das variáveis, como também criar zonas de interação entre o código.

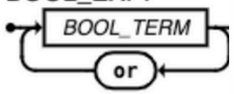
EBNF

```
BLOCK = { STATEMENT } ;

BOOL_EXP = BOOL_TERM, { ("or"), BOOL_TERM } ;

BOOL_TERM = REL_EXP, { ("and"), REL_EXP } ;

REL_EXP = EXPRESSION, { ("below" | "above" | "equal" ) , EXPRESSION,  };

EXPRESSION = TERM, { ("+" | "-" ), TERM } ;

TERM = FACTOR, { ("*" | "/"), FACTOR } ;

STATEMENT =
    (IDENTIFIER, "=", BOOL_EXP
    | "set", IDENTIFIER, ( | ("=", BOOL_EXP))
    | "show", "(", BOOL_EXP, ")"
    | "check", BOOL_EXP, "if", "\n", { ( STATEMENT ) }, [ "else", "\n", { ( STATEMENT ) } ], "
    | "auto", BOOL_EXP, "do", "\n", { ( STATEMENT ) }, "done"
    | "init",IDENTIFIER, "(", IDENTIFIER, ",", IDENTIFIER, ",", IDENTIFIER, ")"), "\n" ;

FACTOR = NUMBER
    | IDENTIFIER,
    | ("+" | "-" | "not"), FACTOR
    | "(", BOOL_EXP, ")" ;

IDENTIFIER = LETTER, { LETTER | DIGIT | "_" };
NUMBER = DIGIT, { DIGIT };
LETTER = ( "a" | "..." | "z" | "A" | "..." | "Z" );
DIGIT = ( "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | "0" );
```

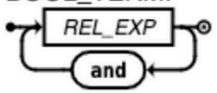
Diagrama Sintático

BLOCK:

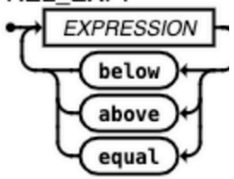
BOOL_EXP:



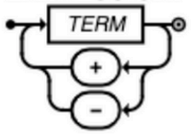
BOOL_TERM:



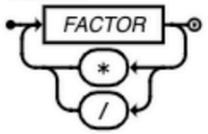
REL_EXP:



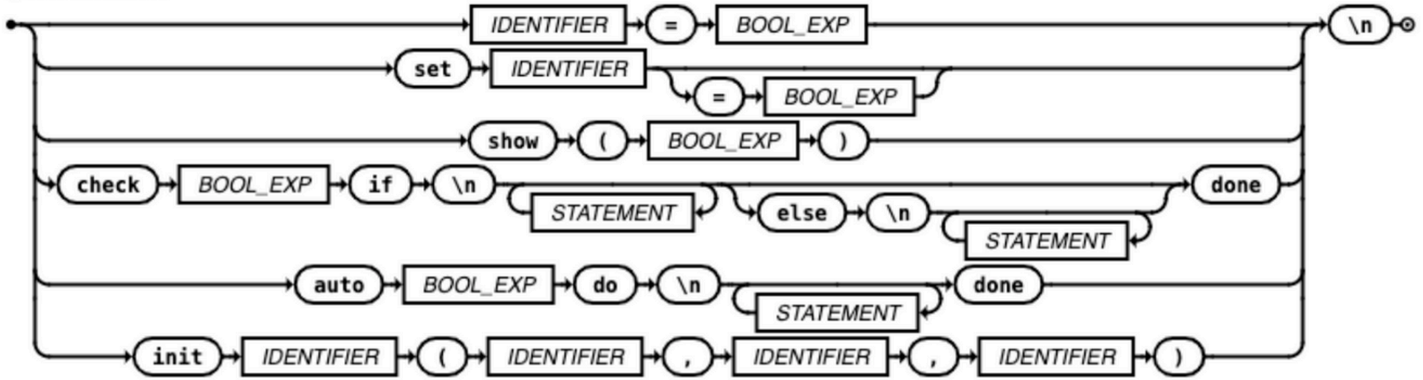
EXPRESSION:



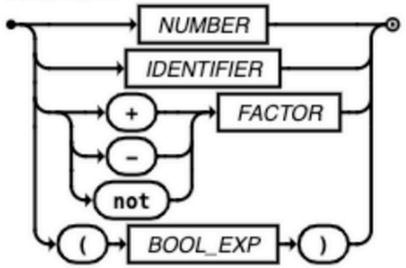
TERM:



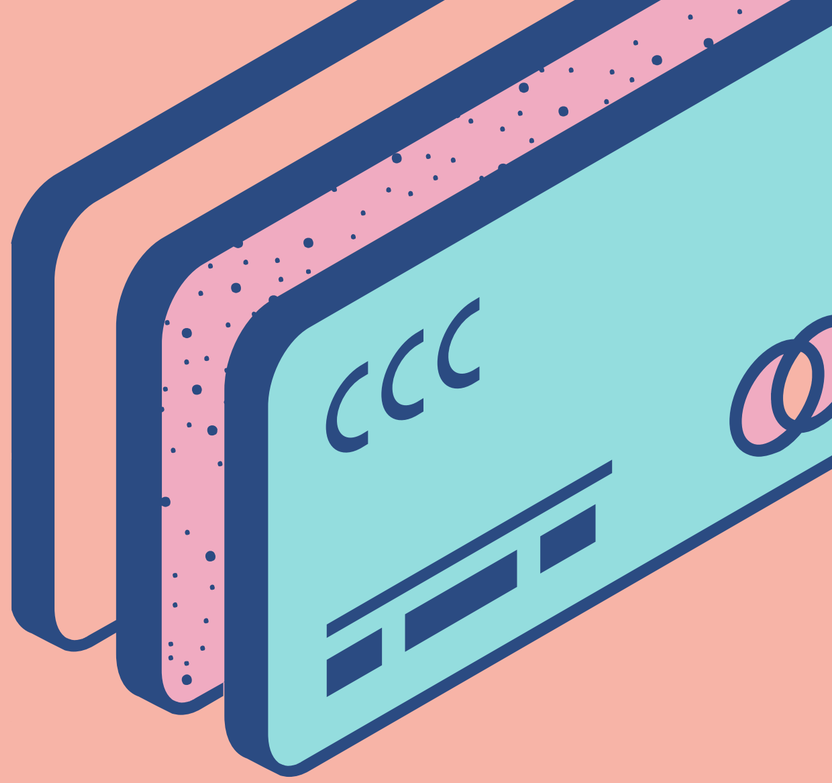
STATEMENT:



FACTOR:



Características



Declaração de Variáveis

Toda declaração de variável é feita através de: "set x", podendo haver agregação de valor ou não.

Mostrar Variáveis

Para realizar a operação de print, é utilizado o token: "show()", que irá avaliar a expressão recebida.

Checagem Condicional

Ao utilizar "check" é criada a estrutura de If/Else, havendo a necessidade de terminar com "done".

Criação de Loops

Para a criação de loops "auto" é passada uma expressão booleana, e após o block interno, há a necessidade de se terminar com "done"

Declaração de Classes

Todas classes são declaradas com "init", e podem ter múltiplos parametros, que serão definidos posteriormente com "nome_classe.parametro_x".

Características

Comentários

Todos os comentários são tratados com o padrão: "--"

Exemplos de Códigos

```
init room_temperature(external, ambient)
init air_conditioner(power, status, target_temperature)

show(room_temperature)
show(air_conditioner)
```

Este código faz a declaração de duas classes, room_temperature e air_condition, respectivamente e mostra todos os parametors declarados, neste caso, todos estarão inicializados com o valor null.

Exemplos de Códigos

```
room_temperature.external = 30
room_temperature.ambient = 30

show (room_temperature)

air_conditioner.power = 1
air_conditioner.status = 0 -- 1 = on, 0 = off
air_conditioner.target_temperature = 10
```

Para dar valor as variáveis/parametros das classes, há necessidade de chamar sempre a classe, e após se referir a variável utilizando ".".

Exemplos de Códigos

```
check room_temperature.external equal room_temperature.ambient if
  air_conditioner.status = 1
else
  check room_temperature.external above room_temperature.ambient if
    air_conditioner.status = 1
  else
    air_conditioner.status = 0
  done
done
```

É possível realizar operações lógicas com as variáveis das classes, sempre se referindo a telas utilizando o prefixo ".".

Exemplos de Códigos

```
check room_temperature.external equal room_temperature.ambient if
  air_conditioner.status = 1
else
  check room_temperature.external above room_temperature.ambient if
    air_conditioner.status = 1
  else
    air_conditioner.status = 0
  done
done
```

É possível realizar operações lógicas com as variáveis das classes, sempre se referindo a telas utilizando o prefixo ".".

Exemplos de Códigos

```
air_conditioner.status = 0
auto ((room_temperature.external above room_temperature.ambient) or (room_temperature.external equal room_temperature.ambient)) and not(air_conditioner.target_temperature equal room_temperature.ambient) do
  air_conditioner.status = 1
  energy_consumption = 1
  room_temperature.ambient = room_temperature.ambient - (1 * air_conditioner.power)
  show(room_temperature.ambient)
done
```

Como também é possível construir estruturas de lógica booleana complexas, e utilizar elas com condição de saída de Loops, como é o caso deste código, que está checando a temperatura de um ambiente.