

# Linguagem de programação do zero

StaticPy

```
mirror_mod = modifier_ob.  
set mirror object to mirror.  
mirror_mod.mirror_object =  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
  
selection at the end -add  
ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.  
mirror_ob.select = 0  
= bpy.context.selected_object  
data.objects[one.name].select  
print("please select exactly  
  
-- OPERATOR CLASSES ----  
  
types.Operator):  
X mirror to the selected  
object.mirror_mirror_x"  
mirror X"  
  
context):  
context.active_object is not
```

# Conteúdo

01

Introdução

02

Motivação

03

Características

04

Exemplos





01

Introdução

# Introdução

## Ideia

Desenvolver uma linguagem de programação do zero, desde a sintaxe (representada pelo EBNF) até compilador completo com Lexer, Parser, Semântico e Geração de Código, feitos com respectivamente: Flex, bison e LLVM.

## Features

A linguagem desenvolvida inclui variáveis, condicionais, loops, funções além de algumas funções built-in.





02

Motivação



# Motivação

## Inspirações

Inspirado em múltiplas linguagens:

- Python
- Rust
- C
- C#

## Objetivo

Criar uma linguagem de alto nível como o python, porém com tipagem estática para prevenir erros em runtime.



03

# Características



# Características

## Bloco de código

Assim como no python, um bloco de código é definido por “:”, seguido de código indentado.

## Built-ins

Temos algumas funções presentes por padrão na linguagem, por exemplo o “println” que funciona como o “printf” do C.

## Tipagem

Em contrapartida ao python, a linguagem exige tipagem estática, porém temos a keyword “var” para inferir o tipo da variável.

## Condicionais

Funcionam como o if/else do python, onde temos “if” seguido da expressão e bloco de código.

## Loops

Assim como no python, temos while e for, porém o for segue uma sintaxe semelhante à do C.

## Funções

Funções seguem o estilo do Rust.



04

Exemplos



# Exemplos - Variáveis e operações

```
int x = 0
var y = 5
x += 5
println("x = %d", x)
println("y = %d", y)
println("x * y = %d", x * y)
println("x / y = %d", x / y)
double z = 5.0
println("z = %lf", z)
println("x / z = %lf", x / z)
println("x % y = %d", x % y)
```

Esse código demonstra a criação de variáveis e algumas das operações aritméticas presentes na linguagem



# Exemplos - Funções, loops e condicionais

```
# Function to calculate the square of a number
```

```
fn square(x: int) -> int:
```

```
    return x * x
```

```
# Main program
```

```
var num = 5
```

```
var result = 0
```

```
# For loop to calculate the sum of squares
```

```
for int i = 0; i <= num; i++:
```

```
    if i > 0:
```

```
        result = result + square(i)
```

```
println("The sum of squares is %d", result)
```

```
# Conditional statement to check if the result is even or odd
```

```
if result % 2 == 0:
```

```
    println("The sum of squares is even.")
```

```
else:
```

```
    println("The sum of squares is odd.")
```

Este código demonstra a criação de uma função para cálculo de potência de dois, e faz uso do for loop para somar o quadrado dos números de 1 até num, e usa condicionais para imprimir se a soma resultante é par ou ímpar.



# Exemplos - Built-ins

```
# Random number generation!!!  
double random  
for int i = 0; i < 10; i++:  
    random = rand()  
    println("random number is: %lf", random)
```

```
# We can time our code  
double start = time()
```

```
for int i = 0; i < 100000; i++:  
    if i % 5000 == 0:  
        println("i = %d", i)
```

```
double end = time()  
println("Time taken: %lf", end - start)
```

Este código demonstra o uso da função “rand” para criar números aleatórios e da “time” para pegar o tempo atual, e assim poder calcular tempo de execução do algoritmo.



# Exemplos - Built-ins

```
int x_int = 25
double y = 3.0

# Must be a double to use sqrt, pow, sin, cos
double x = int_to_double(x_int)
double z = sqrt(x)
println("sqrt(%lf) = %d", x, double_to_int(z)) # print as int
z = pow(x, y)
println("%lf ^ %lf = %lf", x, y, z)
z = sin(x)
println("sin(%lf) = %lf", x, z)
z = cos(x)
println("cos(%lf) = %lf", x, z)
```

Este código demonstra o uso de algumas funções matemáticas presentes na linguagem, como `sen`, `cos`, `sqrt` e `pow`. Além de funções de conversão entre tipos como a conversão de `double` para `int` e vice-versa.