**LAMBDA CALCULUS**

**Sintaxe:**

**.x**  onde:

**=> Função**

**=> Parâmetro**

**X => Corpo (retorno)**

O ponto (.) separa a declaração da função do seu escopo (corpo)

Essa sintaxe se chama **abstração da função**

**Outras sintaxes:**

<exp>: := <var>

<exp>: := <exp> <exp> **Aplicação**

<exp>: := <var>.<exp> **Abstração**

<exp>: := (<exp>) **Agrupamento**

**Abstração:**

|  |  |
| --- | --- |
| **Lambda** | **Javascript** |
| a.b | a => b |
| a.b x | a => b(x) |
| a. (b x) | a => (b(x)) |
| (a.b) x | (a => b)(x) |
| a.b.a | a => b => a **(currying)** |

**Aplicação:**

|  |  |
| --- | --- |
| **Lambda** | **Javascript** |
| f a | f(a) |
| f a b | f(a)(b) **(currying)** |
| (f a) b | (f(a))(b) |
| f(a b) | f(a(b)) |

**Beta-Reduction:**

((a.a)b.c.b) (x) e.f

(b.c.b) (x) e.f

c.x) e.f

X (resultado final)

**Simplificação:**

ab.a b => ab.a b (**bound variables** **(parâmetros ligados))**

Em javascript, a simplificação seria resumida da seguinte forma:

a => b => a(b)