



Plantarium

Uma Linguagem de Programação Inspirada no Ciclo de Calvin

Marlon Silva Pereira

Motivação para essa linguagem



Interesse por Biologia

01. Meu interesse pelo funcionamento dos processos biológicos, especialmente com relação à botânica

Interdisciplinaridade

02. Possibilidade de explorar como linguagens de programação podem simular processos naturais, representando diversas equações bioquímicas através de equações, loops e condicionais



Características da linguagem

01.

Variáveis padrões

Variáveis que precisam ser declaradas para a execução do ciclo

02.

Variáveis personalizadas

Variáveis opcionais que podem ser declaradas para auxiliar na programação

03.

Prints

Prints para exibir resultados e verificar as saídas dos ciclos

04.

Ciclo de Calvin

Bloco dedicado para o ciclo, com suas principais funções e possibilidade de customização

05.

Condicionais

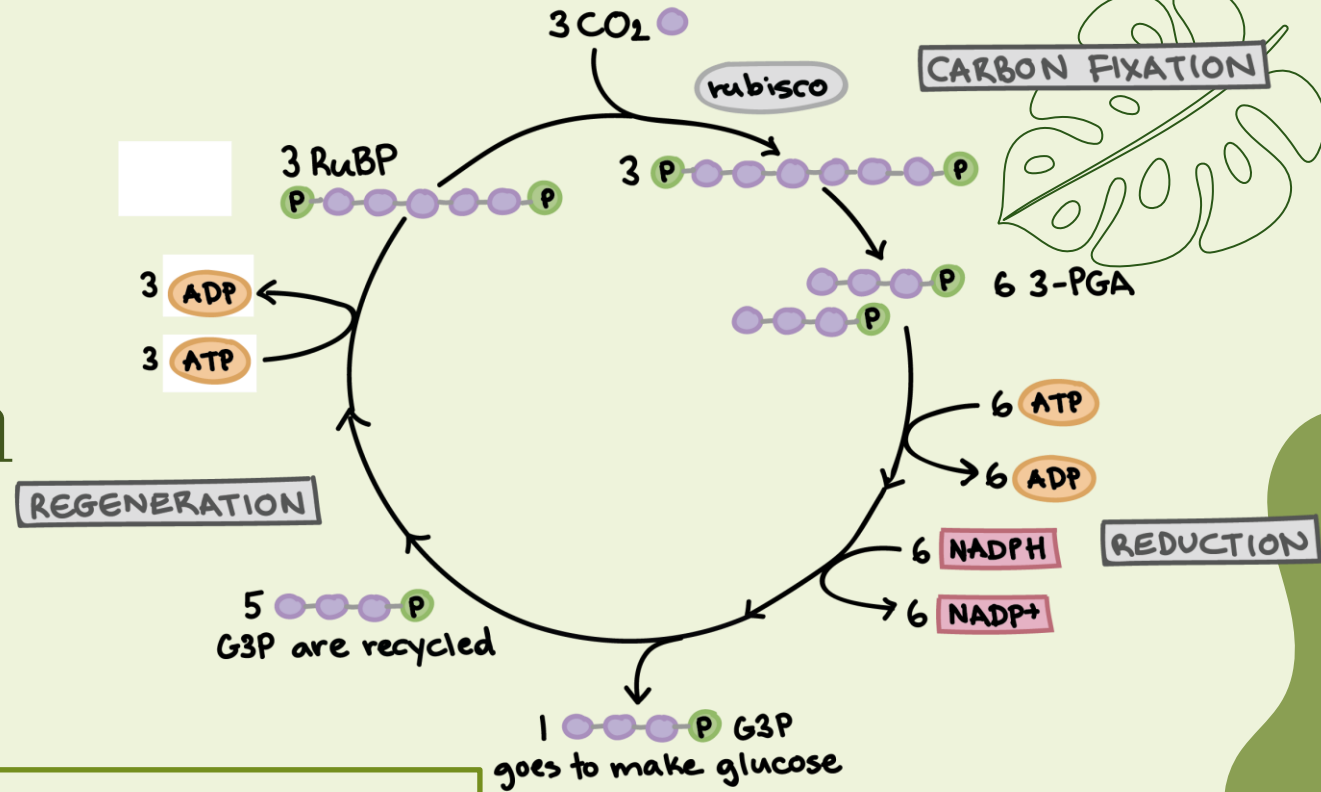
Condicionais e operações para comparar selecionar os caminhos de execução

06.

Recursividade

O ciclo principal pode ser chamado recursivamente para observar consecutivas execuções

Ciclo de Calvin



$\text{CO}_2 + \text{RuBP} \rightarrow 2 \text{ 3-PGA}$

$3\text{-PGA} + \text{NADPH} + \text{ATP} \rightarrow \text{G3P} + \text{NADP}^+ + \text{ADP} + \text{P}_i$

$5 \text{ G3P} + 3 \text{ ATP} \rightarrow 3 \text{ RuBP} + 3 \text{ ADP}$

$1 \text{ G3P} \rightarrow 1/2 \text{ Glucose}$

EBNF

```
program = { statement };
```

```
statement = variable_declaration  
           | custom_variable_declaration  
           | assignment_statement  
           | operation_statement  
           | cycle_of_calvin  
           | print_statement;
```

```
variable_declaration = "variable", (RuBP | glicose | RuBP_min | glicose_min), "=",  
number, ";"; // Declaração de variáveis padrão
```

```
custom_variable_declaration = "custom_variable", identifier, ";"; // Declaração de  
variáveis personalizadas
```

```
assignment_statement = (identifier | custom_variable), assignment_operator,  
(identifier | custom_variable | number), ";"; // Declaração de atribuição
```

```
operation_statement = (identifier | custom_identifier), assignment_operator,  
(identifier | custom_identifier | number), arithmetic_operator, (identifier |  
custom_identifier | number), ";"; // Declaração de operação
```

```
cycle_of_calvin = "calvin_cycle", "{", statements, "fixação_CO2", "redução_3_PGA",  
conditionals, statements, "}"; // Ciclo de Calvin com condições
```

```
statements = { assignment_statement | operation_statement };
```

```
conditionals = if_statement, else_statement;
```

```
if_statement = "if", "(", condition, ")", "{", function, "}";
```

```
else_statement = "else", "{", function, "}";
```

```
function = "regeneração_RuBP", "síntese_glicose";
```

```
condition = identifier, comparison_operator, number  
           | identifier, logical_operator, identifier  
           | "(", condition, ")", logical_operator, "(", condition, ");"
```

```
assignment_operator = "=" | "+=" | "-=" | "*=" | "/="; // Operadores de atribuição
```

```
arithmetic_operator = "+" | "-" | "*" | "/"; // Operadores aritméticos
```

```
comparison_operator = "==" | "!=" | "<" | ">" | "<=" | ">=";
```

```
logical_operator = "and" | "or";
```

```
print_statement = "print", "(", string_literal | identifier, ")", ";"; // Comando de  
impressão
```

```
identifier = RuBP | glicose | RuBP_min | glicose_min;
```

```
custom_identifier = letter, { letter | digit | "_" };
```

```
RuBP = "RuBP";
```

```
glicose = "glicose";
```

```
RuBP_min = "RuBP_min";
```

```
glicose_min = "glicose_min";
```

```
number = digit, { digit };
```

```
string_literal = "", { all_characters_except_quotes }, "";
```

```
letter = "a" | "b" | ... | "z" | "A" | "B" | ... | "Z";
```

```
digit = "0" | "1" | ... | "9";
```

```
all_characters_except_newline = any_visible_character | space;
```

```
all_characters_except_quotes = all_characters_except_newline | "";
```

```
space = " ";
```

Exemplo de código

```
// Declaração de variáveis padrão
variable CO2 = 20;
variable RuBP = 10;
variable glicose = 10;
variable ATP = 10;
variable ADP = 0;
variable G3P = 10;
variable RuBP_min = 3; // Valor mínimo de RuBP
variable glicose_min = 2; // Valor mínimo de glicose
variable PGA = 0;
variable NADPH = 5;
variable NADP = 0;
variable Pi = 0;

// Declaração de variáveis personalizadas
custom_variable luminosidade = 50;
custom_variable pH = 6;
custom_variable count = 0;



calvin_cycle {
    fixacao_CO2;
    reducao_3_PGA;

    // Condicional para regeneração_RuBP
    if (RuBP < RuBP_min and glicose > glicose_min) {
        regeneracao_RuBP;
    } else {
        sintese_glicose;
    };

    // Condicional para execução de novo ciclo
    if (RuBP > RuBP_min and glicose > glicose_min) {
        count = count + 1;
        pH = pH + 1;
        luminosidade = luminosidade - 2;
        calvin_cycle;
    };
    print("Numero de ciclos: ")
    print(count)
    print("pH final: ")
    print(pH)
    print("luminosidade final: ")
    print(luminosidade)
}
```

Saida:

Numero de ciclos:
6
pH final:
12
luminosidade final:
38



Repositório

https://github.com/marlonsp/APS_LogComp

Referências e imagem: <https://pt.khanacademy.org/science/biology/photosynthesis-in-plants/the-calvin-cycle-reactions/a/calvin-cycle>



CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon** and infographics & images by **Freepik**

