

LÓGICA DE PROGRAMAÇÃO : aula 03

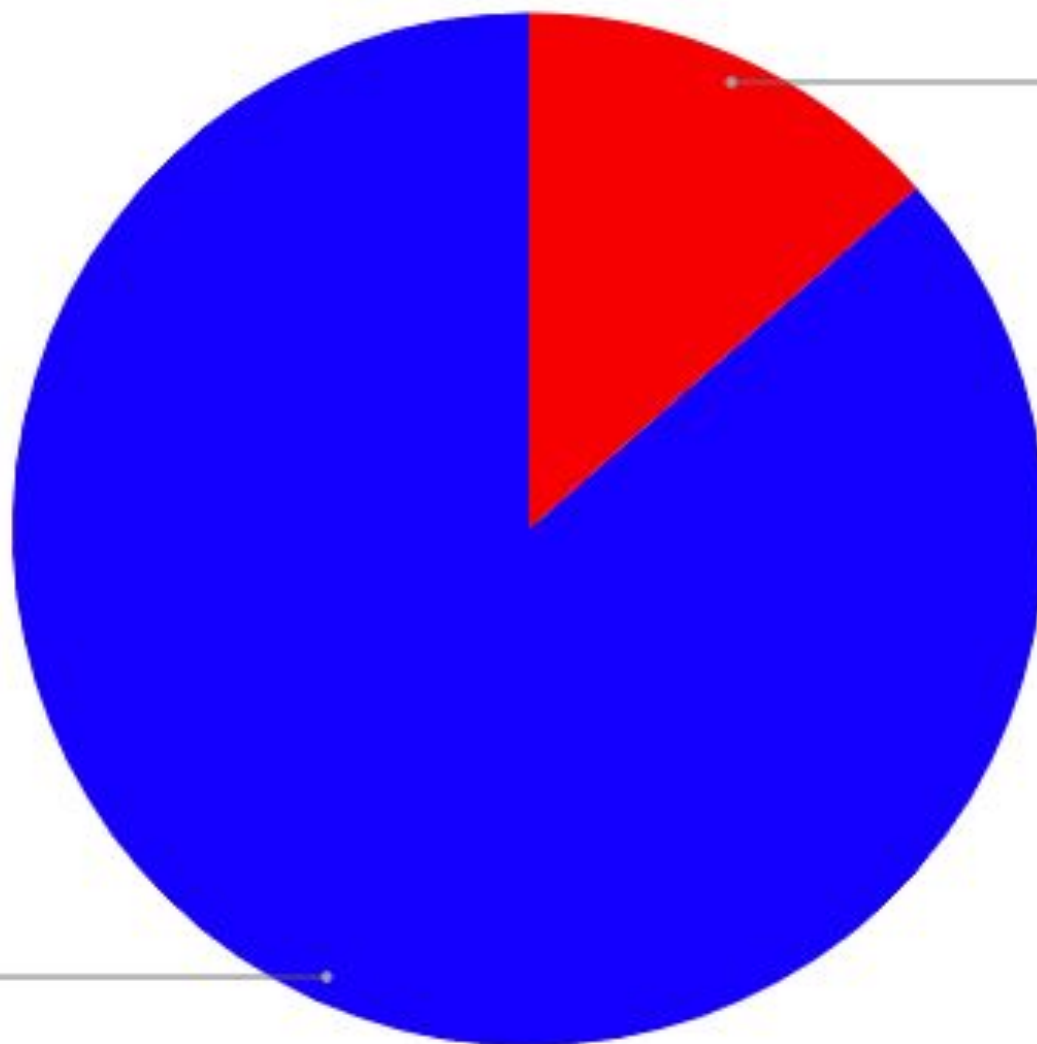
PROFESSOR KALIL DE OLIVEIRA

SECRETARIA DE ESTADO DA
EDUCAÇÃO/SED

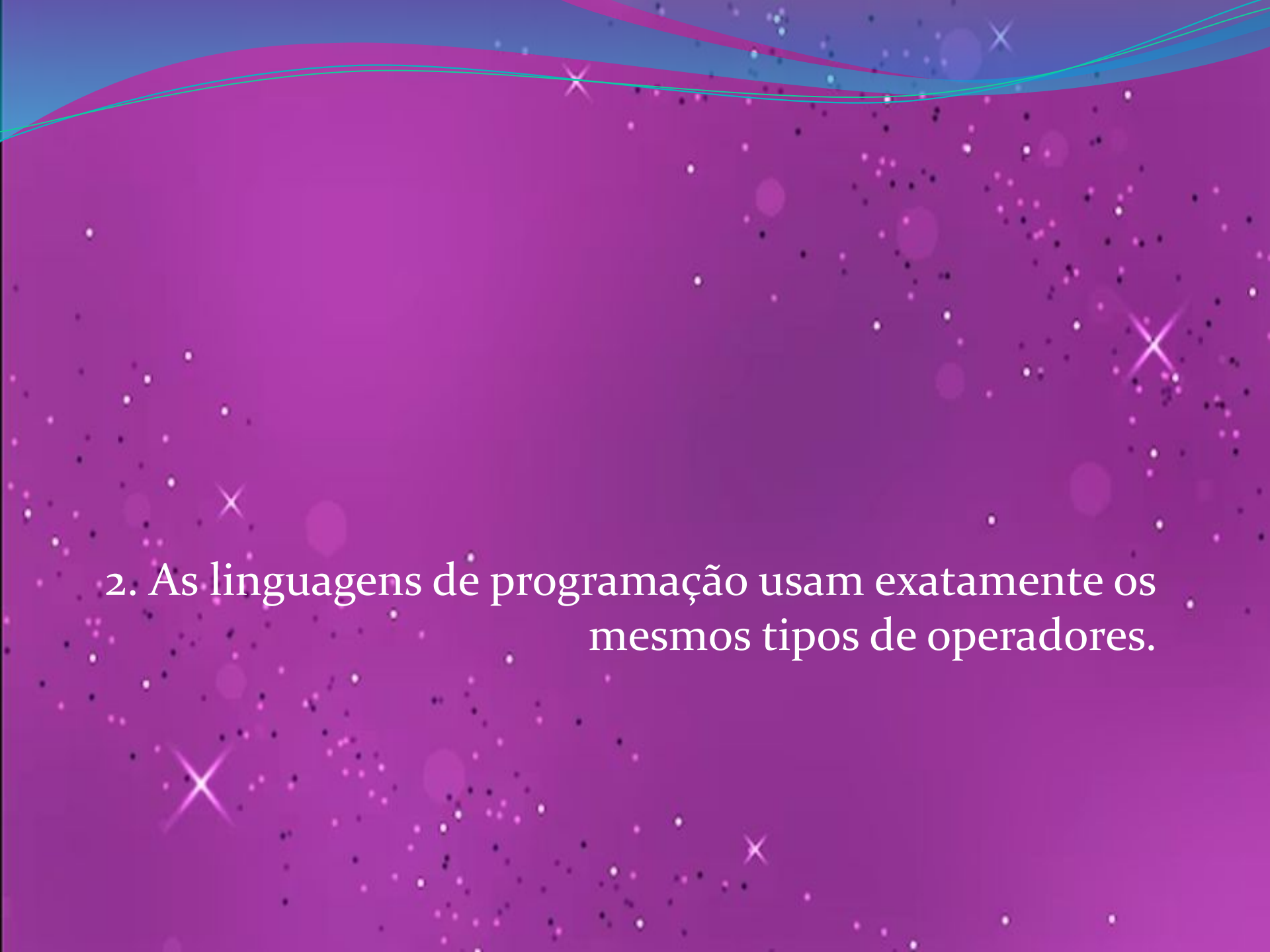
Revisão: Falso ou verdadeiro?

1. Entre outras coisas, operadores existem para atribuírmolos valores às variáveis.

VERDADEIRO
86,5%



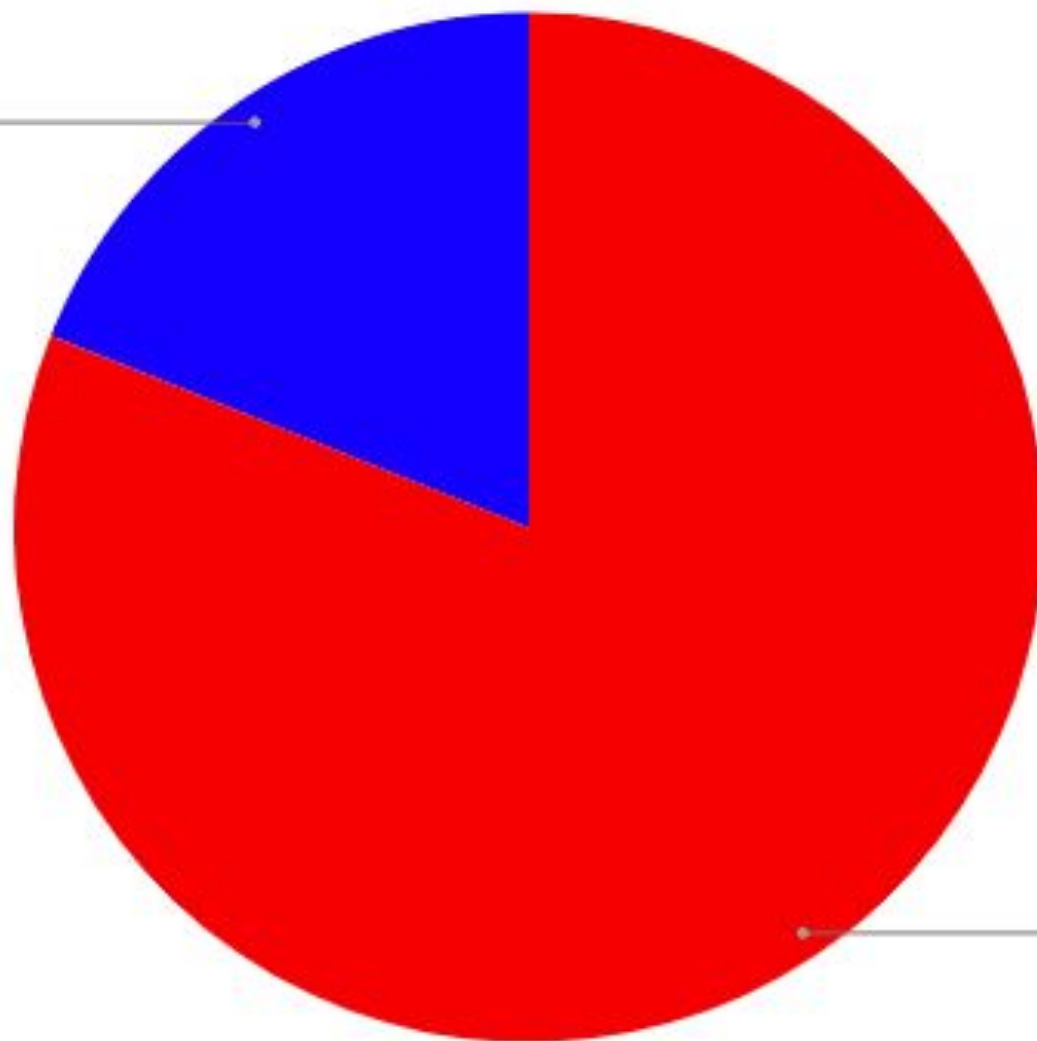
FALSO
13,5%



2. As linguagens de programação usam exatamente os mesmos tipos de operadores.

VERDADEIRO

18,9%



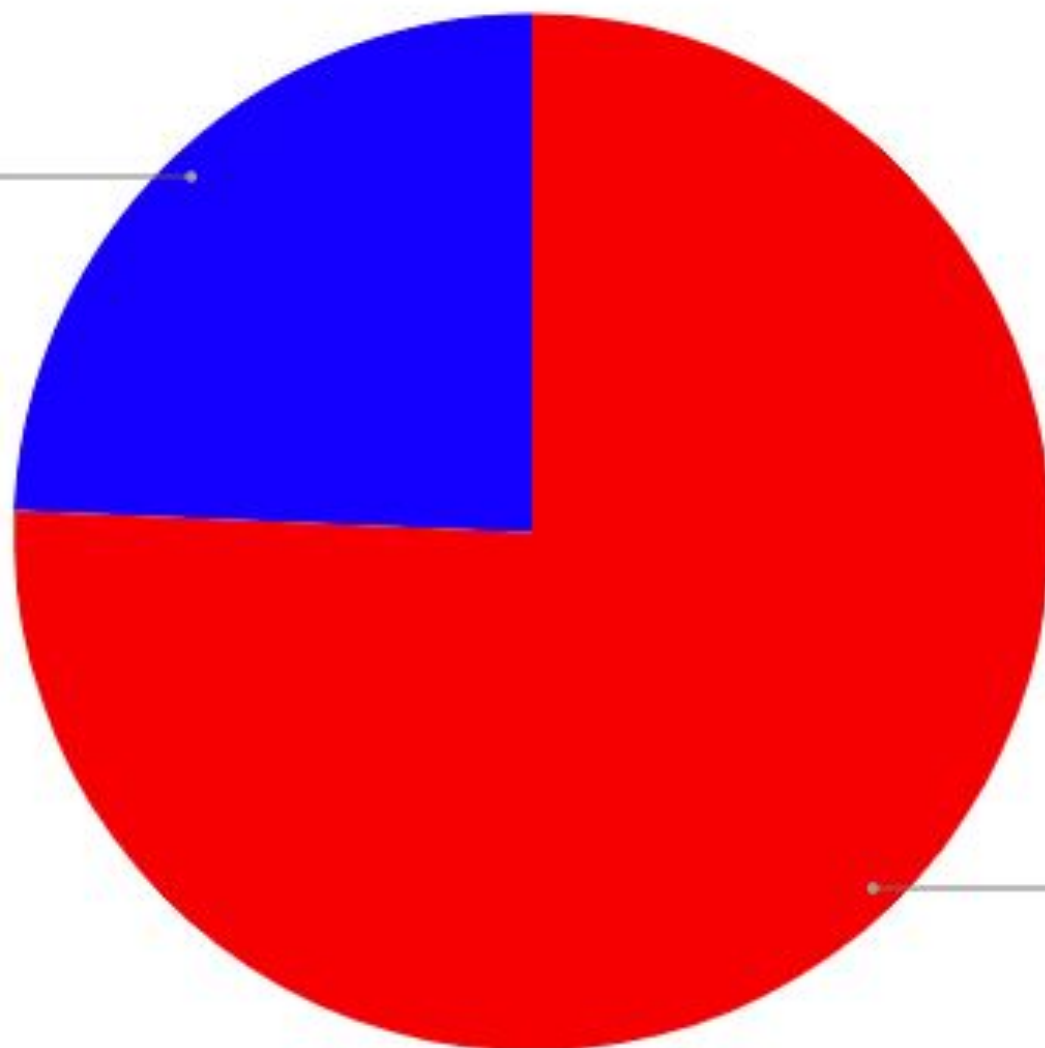
FALSO

81,1%

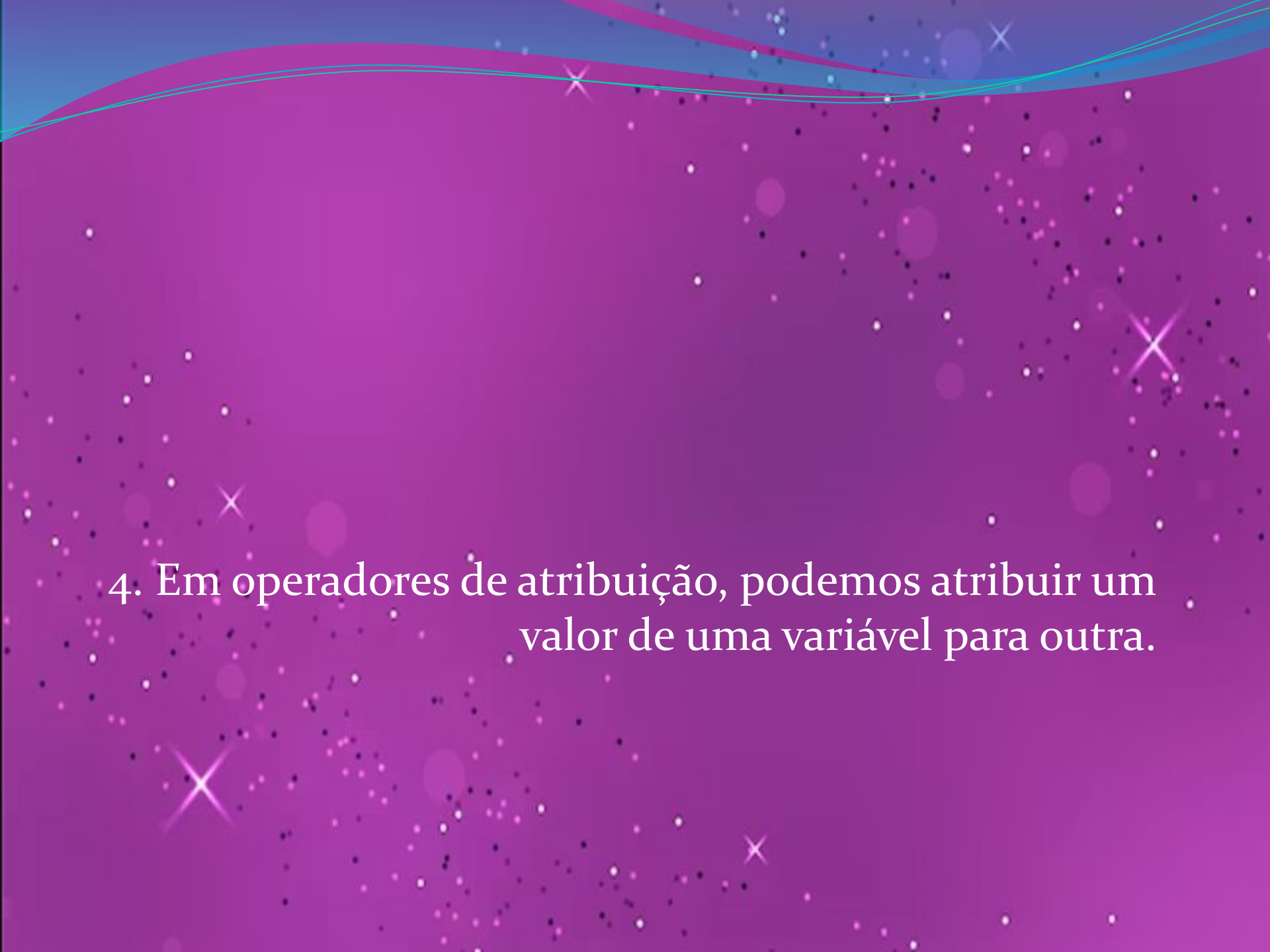


3. Um operador de comparação só funciona com variáveis do tipo booleana.

VERDADEIRO
24,3%

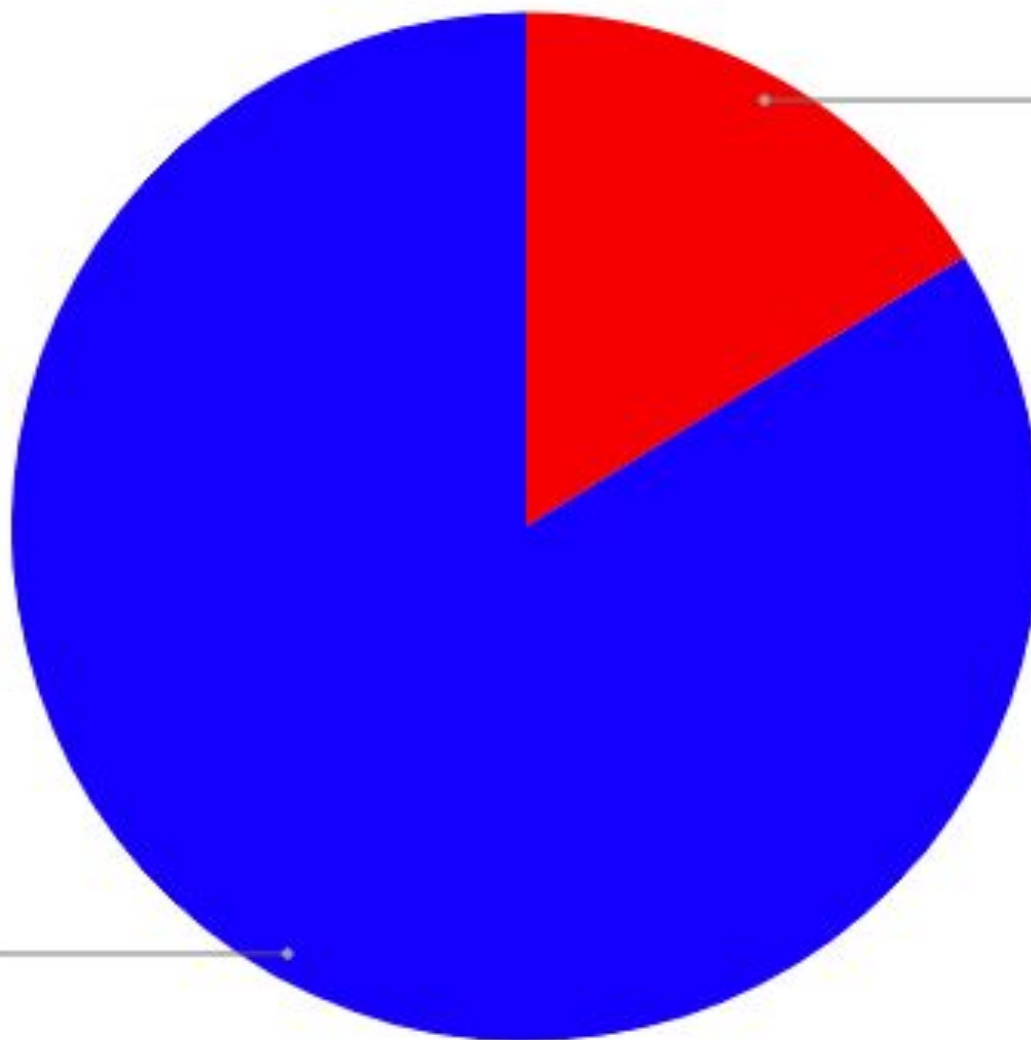


FALSO
75,7%

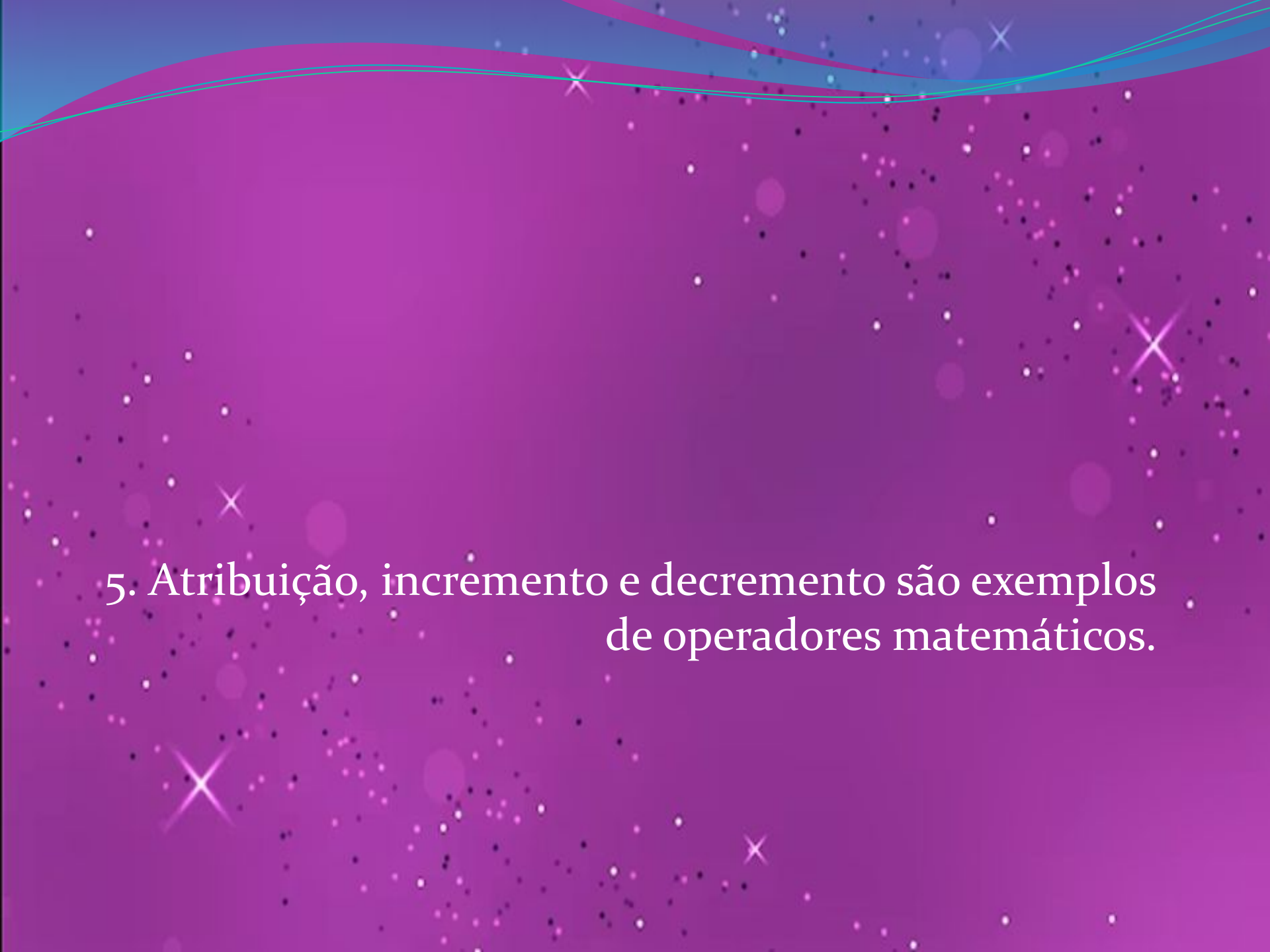


4. Em operadores de atribuição, podemos atribuir um valor de uma variável para outra.

VERDADEIRO
83,8%

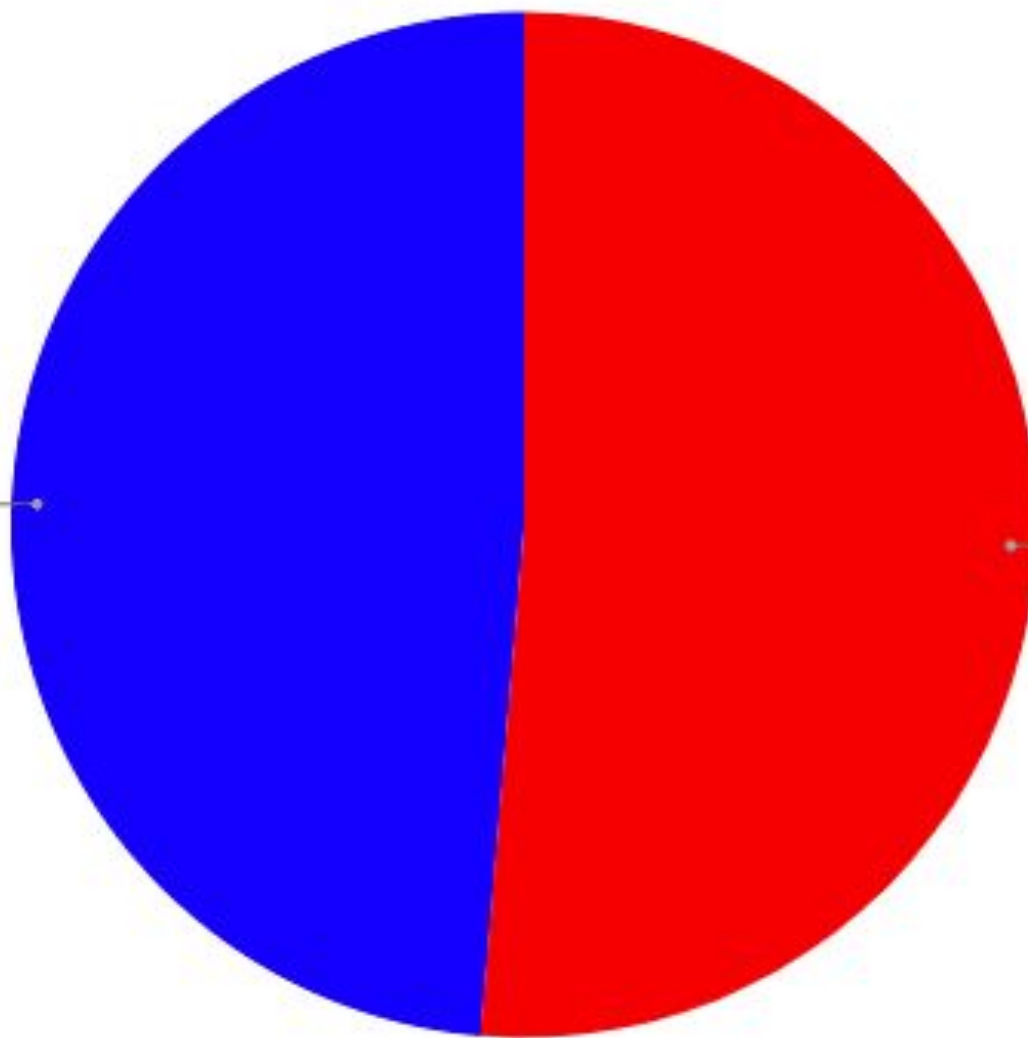


FALSO
16,2%



5. Atribuição, incremento e decremento são exemplos de operadores matemáticos.

VERDADEIRO
48,6%

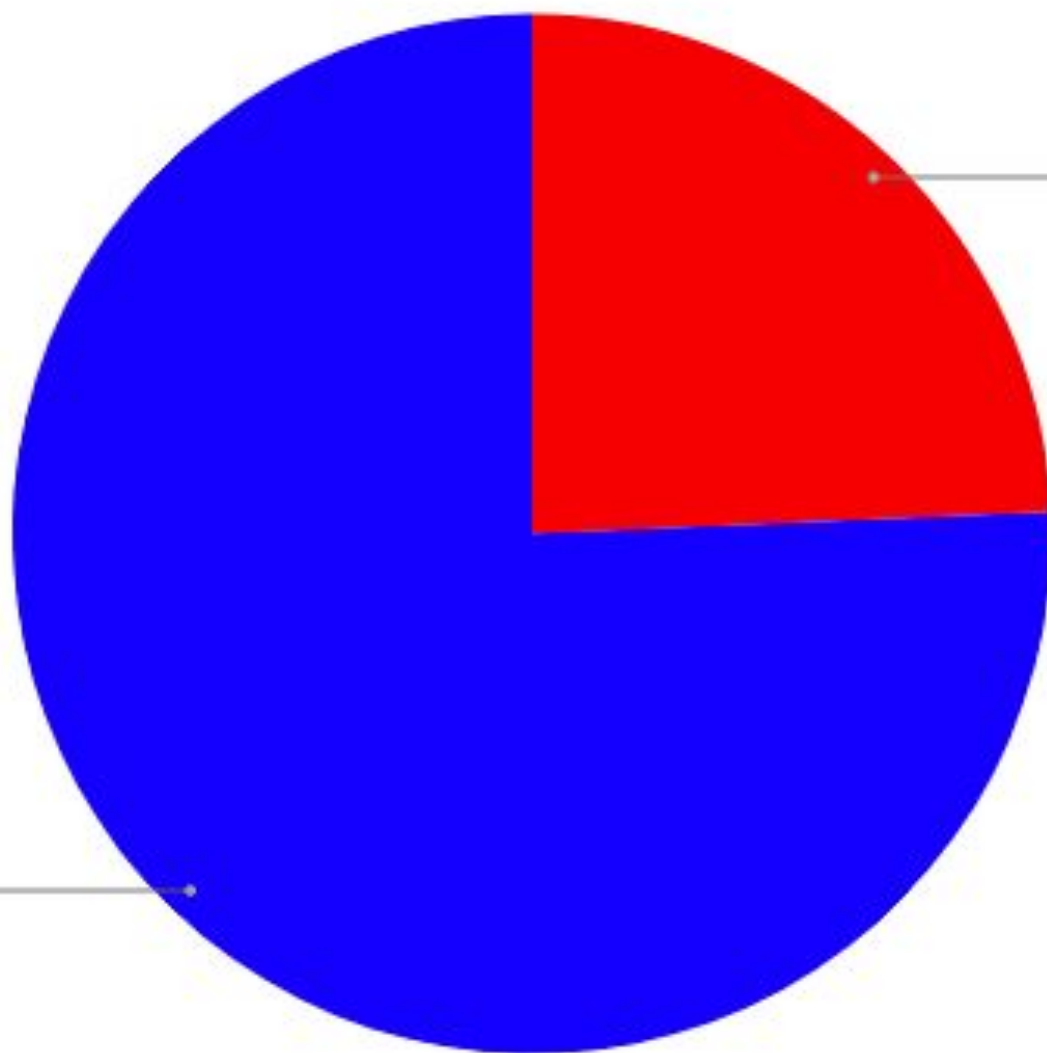


FALSO
51,4%




6. O resto da divisão é um operador matemático útil para descobrir se um número é par.

VERDADEIRO
75,7%

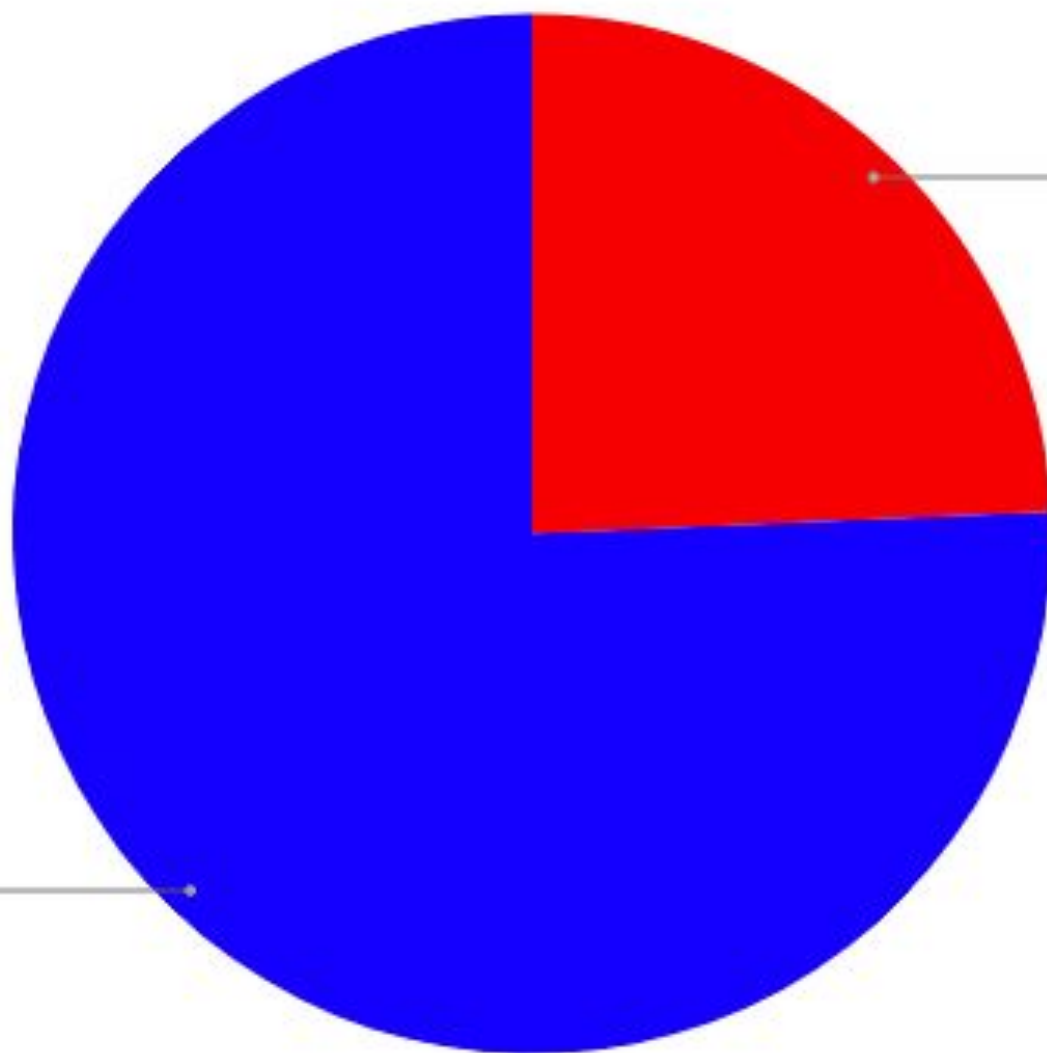


FALSO
24,3%



7. Em JS, na comparação, há diferença entre variáveis terem valores iguais e idênticos.

VERDADEIRO
75,7%

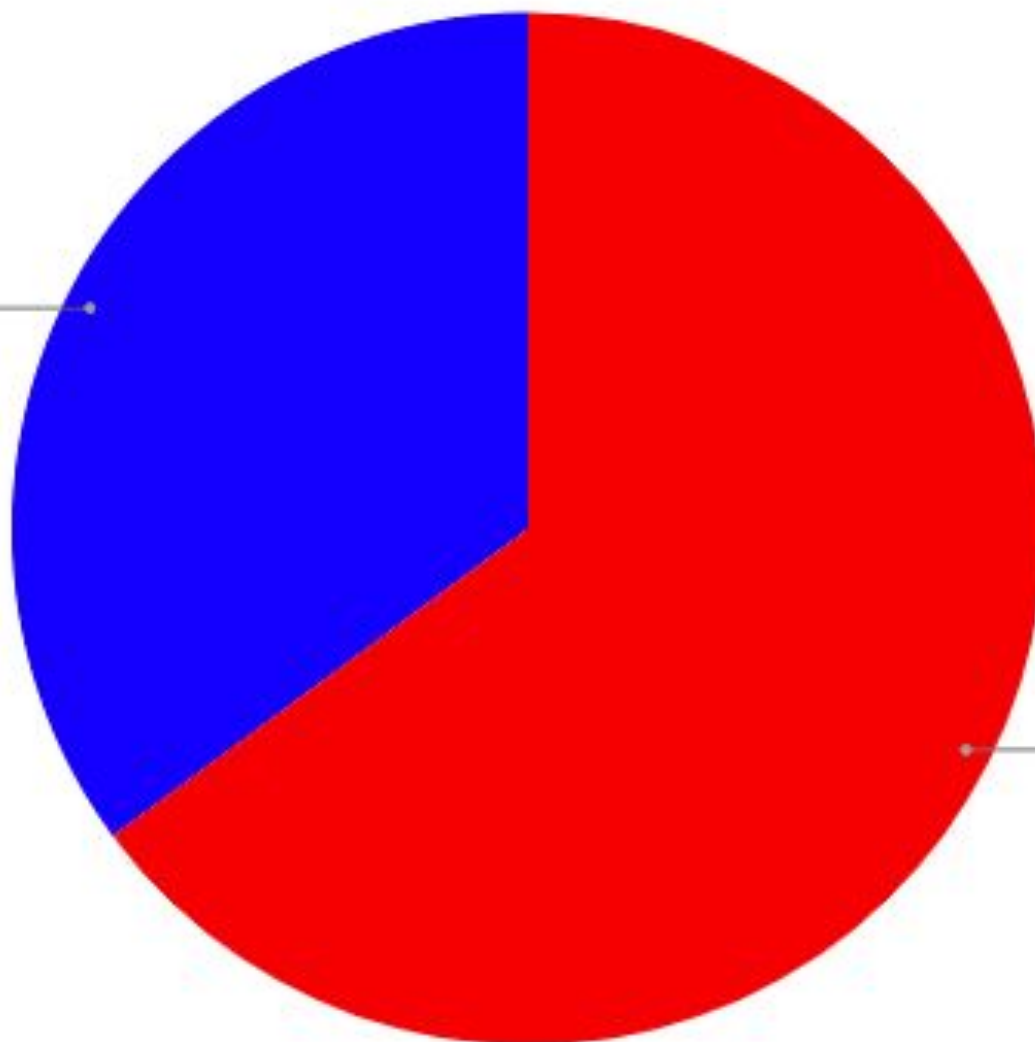


FALSO
24,3%

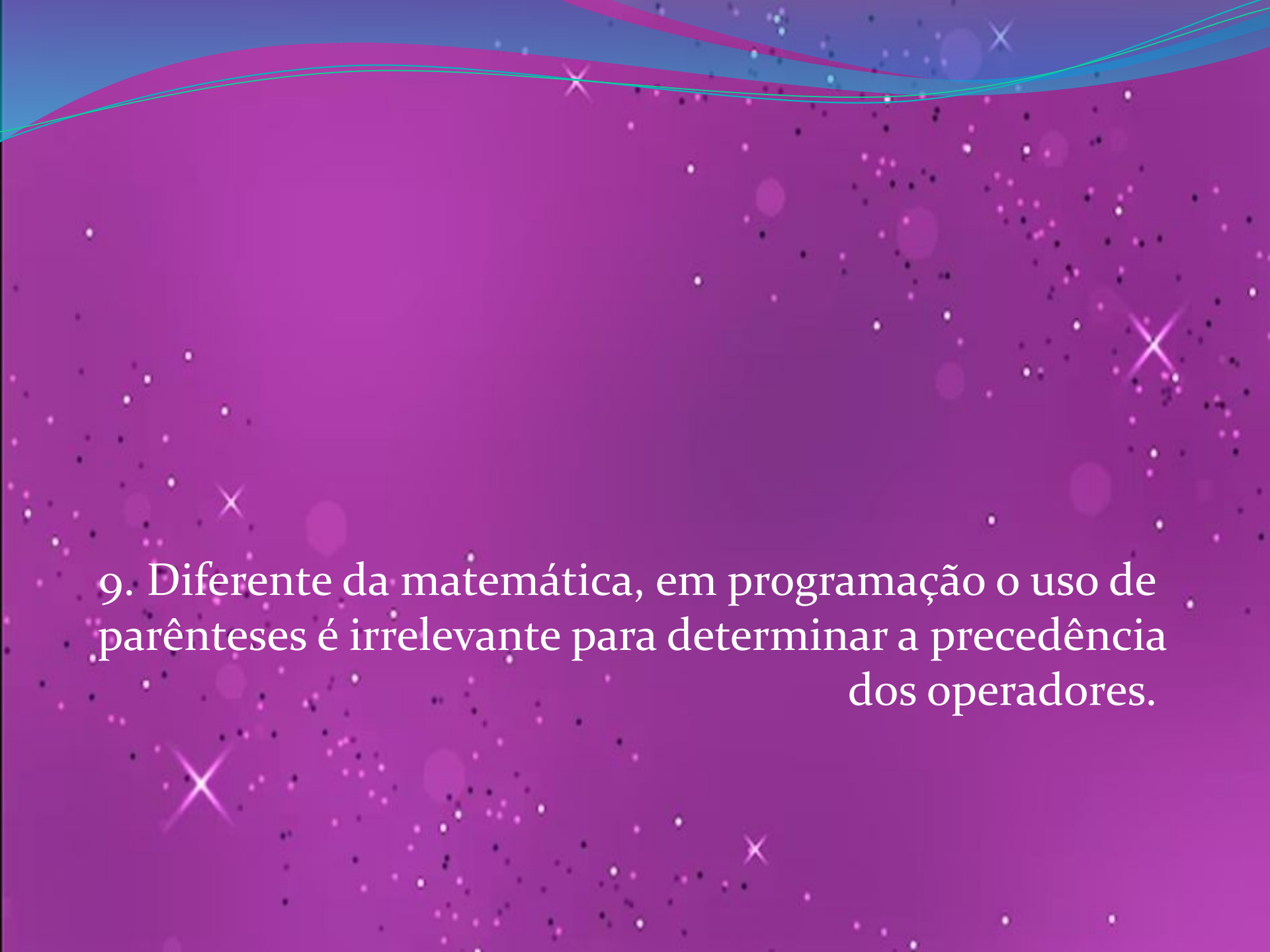


8. A linguagem de programação C não possui variáveis do tipo booleana.

VERDADEIRO
35,1%



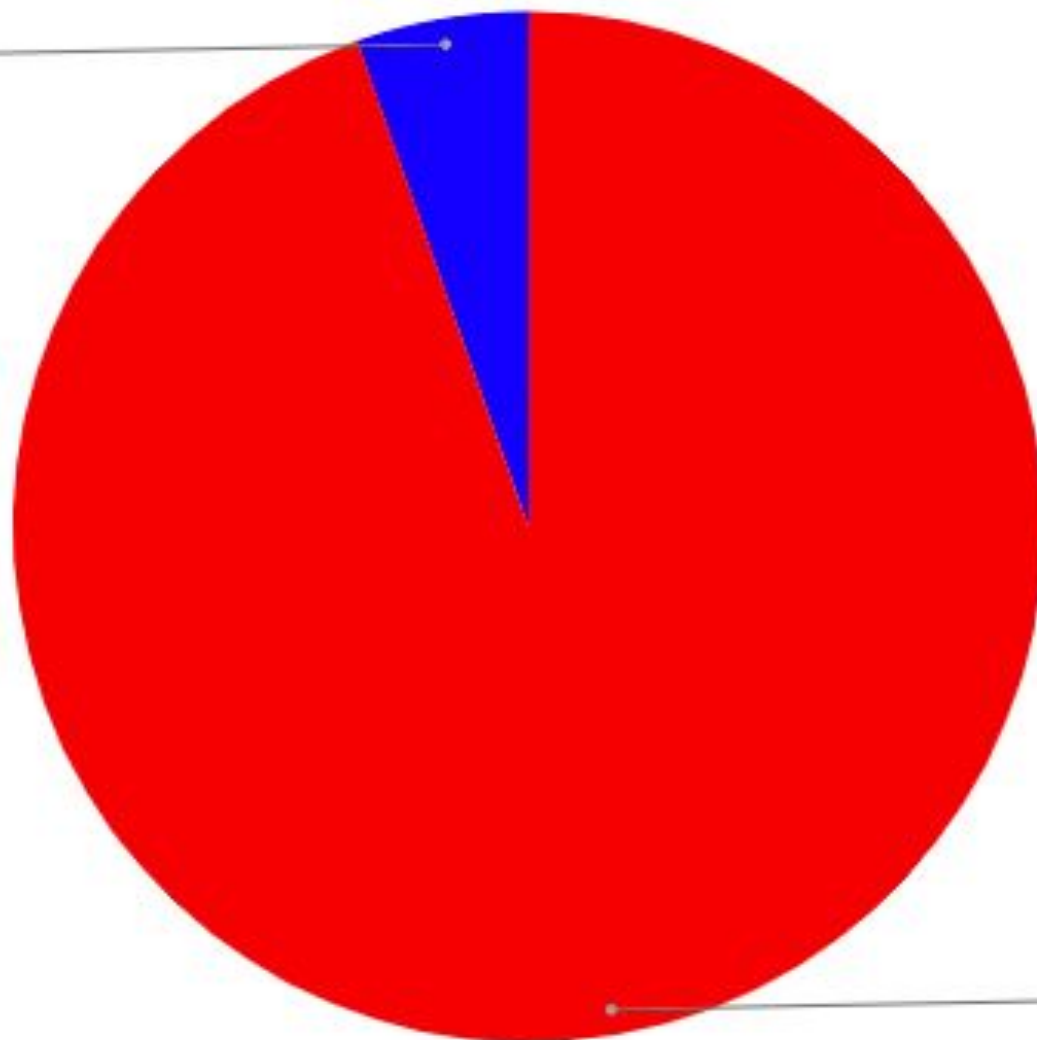
FALSO
64,9%



9. Diferente da matemática, em programação o uso de parênteses é irrelevante para determinar a precedência dos operadores.

VERDADEIRO

5,4%



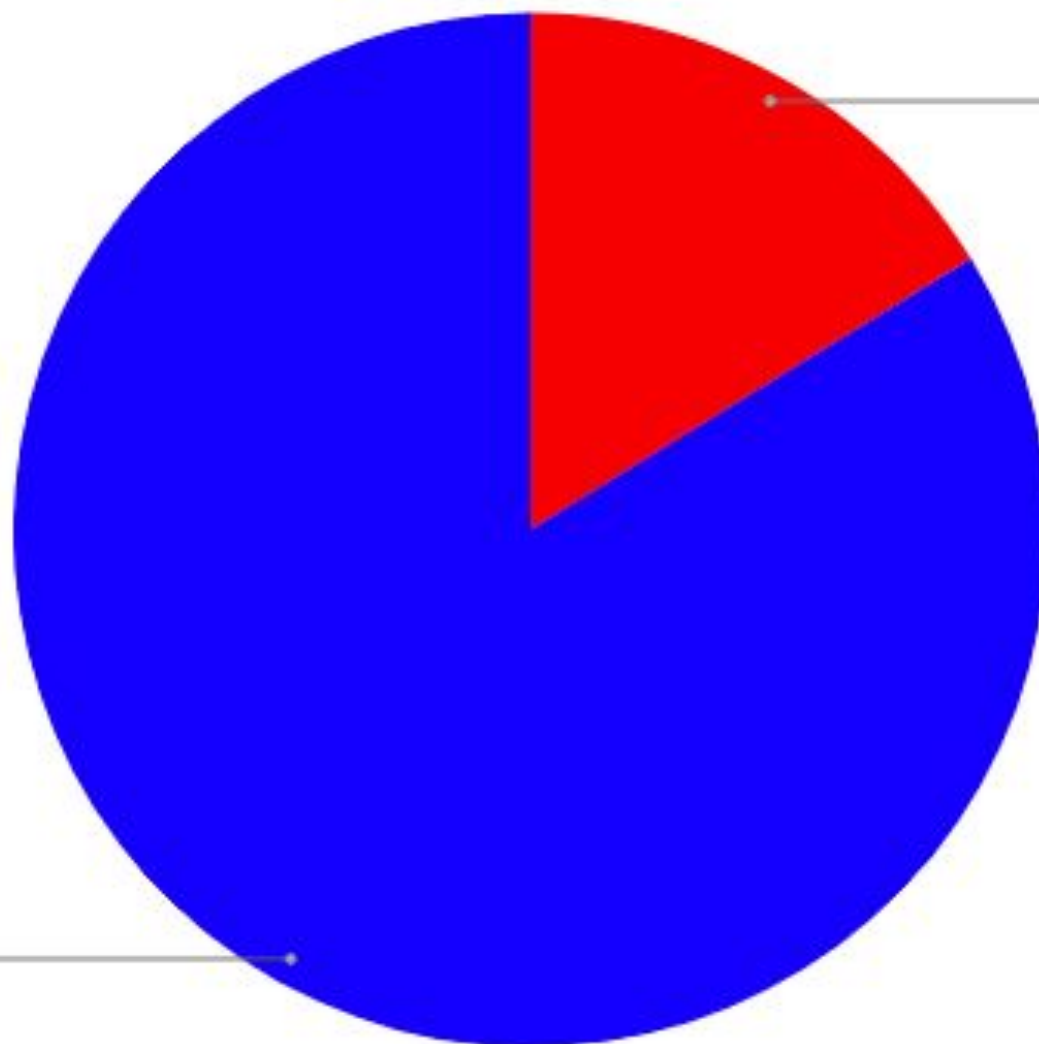
FALSO

94,6%



10. "Maior que" e "menor que" são exemplos de operadores relacionais

VERDADEIRO
83,8%



FALSO
16,2%

Valor das expressões lógicas

Considerando as variáveis A, B, C e D, vamos determinar os valores de algumas expressões lógicas de acordo com os operadores sobre as variáveis, $A=V$, $B=F$, $C=F$ e $D=V$:

Code**Blame**

4 lines (4 loc) · 103 Bytes

```
1      2)  $S = (D \vee A \wedge C) \vee \sim(B \vee C) = v$   
2      3)  $S = \sim(A \vee B) = f$   
3      4)  $S = \sim(A \wedge B) = v$   
4      5)  $S = C \vee \sim B \wedge (A \vee \sim D) = v$ 
```

3 $S = (D \vee A \wedge C) \vee \sim(B \vee C)$

4 $(V \vee V \wedge F) \vee \sim(F \vee F)$

5 $(V \vee V \wedge F) \vee \sim F$

6 $(V \wedge F) \vee \sim F$

7 $F \vee V$

8 V

9 segunda

10 $S = \sim(A \vee B)$

11 $\sim(V \vee F)$

12 $F.$

13 terceira

14 $S = \sim(A \wedge B)$

15 $\sim(V \wedge F)$

16 $F.$

17 quarta

18 $S = C \vee \sim B \wedge (A \vee \sim D)$

19 $F \vee \sim F \wedge (V \vee \sim V)$

20 $F \vee \sim F \wedge (V \vee F)$

21 $F \vee V \wedge V$

22 $V \wedge V$

23 V

1 2) $S = (D \vee A \wedge C) \vee \sim(B \vee C)$

2 $S = (V \vee V \wedge F) \vee \sim(F \vee F)$

3 $S = F \vee V$

4 $S = V$

5

6 3) $S = \sim(A \vee B)$

7 $S = \sim(V \vee F)$

8 $S = F$

9

10 4) $S = \sim(A \wedge B)$

11 $S = \sim(V \wedge F)$

12 $S = V$

13

14 5) $S = C \vee \sim B \wedge (A \vee \sim D)$

15 $S = F \vee \sim F \wedge (V \vee F)$

16 $S = V \wedge V$

17 $S = V$

Responda!

Imagine um programa que receba duas entradas e retorne o resultado da soma, subtração, multiplicação e divisão. Neste programa vamos usar que tipo de operador? Pense. Pesquise. Justifique.

Soma

soma_resultado = 10 + 5

print("Soma:", soma_resultado)

Subtração

subtracao_resultado = 10 - 5

print("Subtração:", subtracao_resultado)

Multiplicação

multiplicacao_resultado = 10 * 5

print("Multiplicação:", multiplicacao_resultado)

Divisão

divisao_resultado = 10 / 5

print("Divisão:", divisao_resultado)

```
#include <stdio.h>
```

```
int main() {
```

```
    // Soma
```

```
    int soma_resultado = 10 + 5;
```

```
    printf("Soma: %d\n", soma_resultado);
```

```
    // Subtração
```

```
    int subtracao_resultado = 10 - 5;
```

```
    printf("Subtração: %d\n", subtracao_resultado);
```

```
    // Multiplicação
```

```
    int multiplicacao_resultado = 10 * 5;
```

```
    printf("Multiplicação: %d\n", multiplicacao_resultado);
```

```
    // Divisão
```

```
    int divisao_resultado = 10 / 5;
```

```
    printf("Divisão: %d\n", divisao_resultado);
```

```
    return 0;
```

```
}
```


Assunto novo! Processo e saída...

O que adianta saber que o computador recebe um dado se não souber o que fazer com ele? O que é um processo? Como fazer o dado retornar ao usuário com a informação que ele deseja?

Muitas perguntas!

Não se assuste. Vamos ver como é o processo em programas simples, como uma calculadora de somar.
Aos poucos você vai pegando a prática...

python

```
# Definindo os valores
```

```
num1 = 7
```

```
num2 = 1
```

```
# Realizando a soma
```

```
soma = num1 + num2
```

```
# Exibindo o resultado
```

```
print(f"A soma de {num1} e {num2} é: {soma}")
```

```
#include <stdio.h>
```

```
int main() {
```

```
    // Definindo os valores
```

```
    int num1 = 7;
```

```
    int num2 = 1;
```

```
    // Realizando a soma
```

```
    int soma = num1 + num2;
```

```
    // Exibindo o resultado
```

```
    printf("A soma de %d e %d é: %d\n", num1, num2, soma);
```

```
    return 0;
```

```
}
```

Vamos ao replit.com

É possível acessar o ambiente de desenvolvimento on-line (também chamado de I.D.E.) em qualquer navegador digitando <https://repl.it>

Sign Up

Log In



Clicar em “*Sign Up*” e fazer o login
com a sua conta do github.



+ Create Repl



Home



My Repls



Deployments



Usage



Teams

Você entrou e agora só precisa clicar em “**Create Repl**” e escolher a linguagem que pretende desenvolver seu aplicativo

Create a Repl

Import from GitHub

Template

Python

Title

somar_um_numero



Languages

Public

Anyone can view and fork this Repl.

Python ✓

Python is a high-level, interpreted, general-purpose programming language.



replit

♥ 4.1K + 47.5M

⚡ Upgrade to make private

+ Create Repl

No nosso caso, escolhemos Python. Nosso projeto vai se chamar “somar_um_numero”. Se acostume a usar este tipo de escrita em Python. Clicar em **Create Repl**

Nosso código ficou assim...

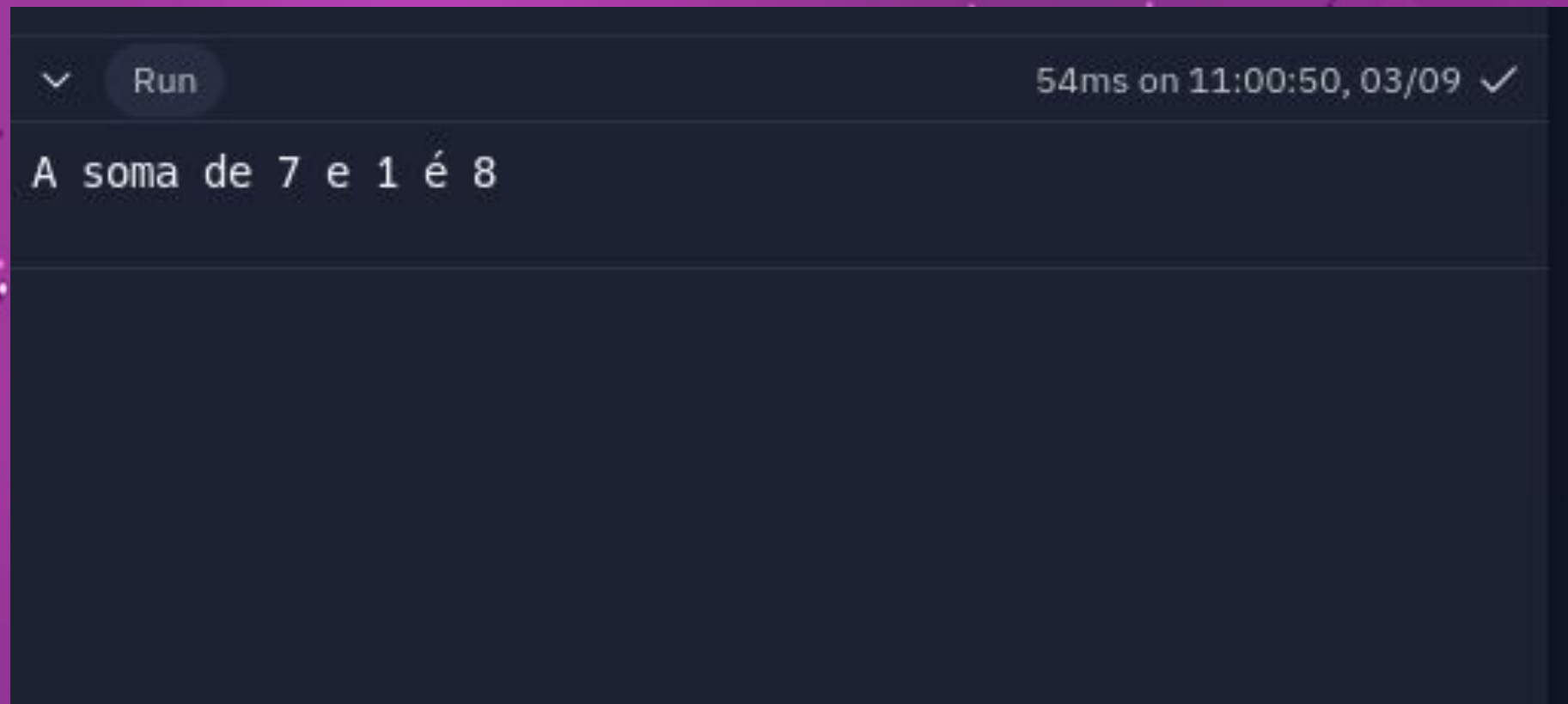


main.py > ...

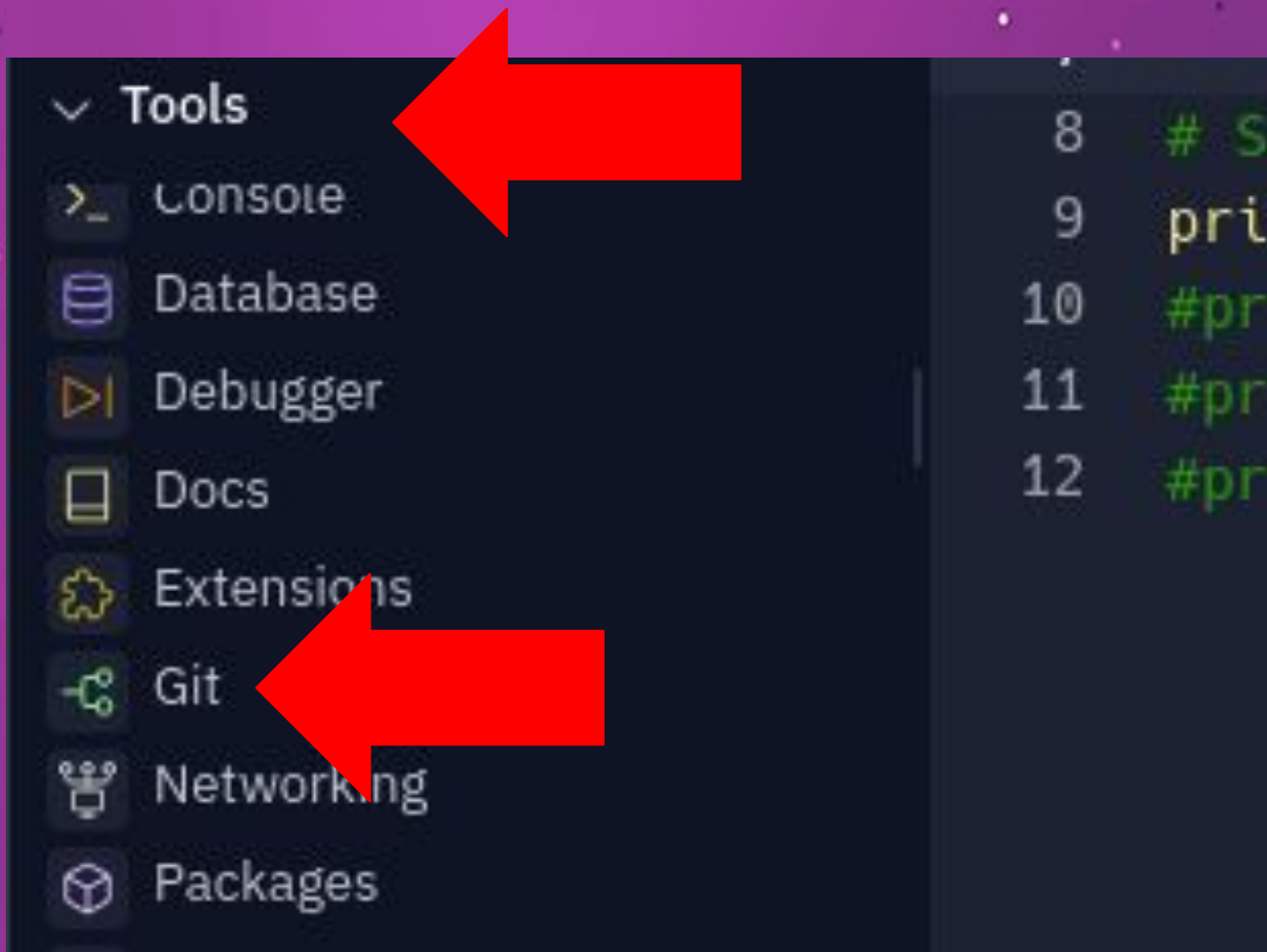
Format

```
1  # Variaveis e valores
2  num1 = 7
3  num2 = 1
4
5  # Processos
6  soma = num1 + num2
7
8  # Saída
9  print(f"A soma de {num1} e {num2} é {soma}")
10 #print("A soma de {} e {} é {}".format(num1, num2, soma))
11 #print("A soma de", num1, "e", num2, "é", soma)
12 #print("A soma de " + str(num1) + " e " + str(num2) + "
```

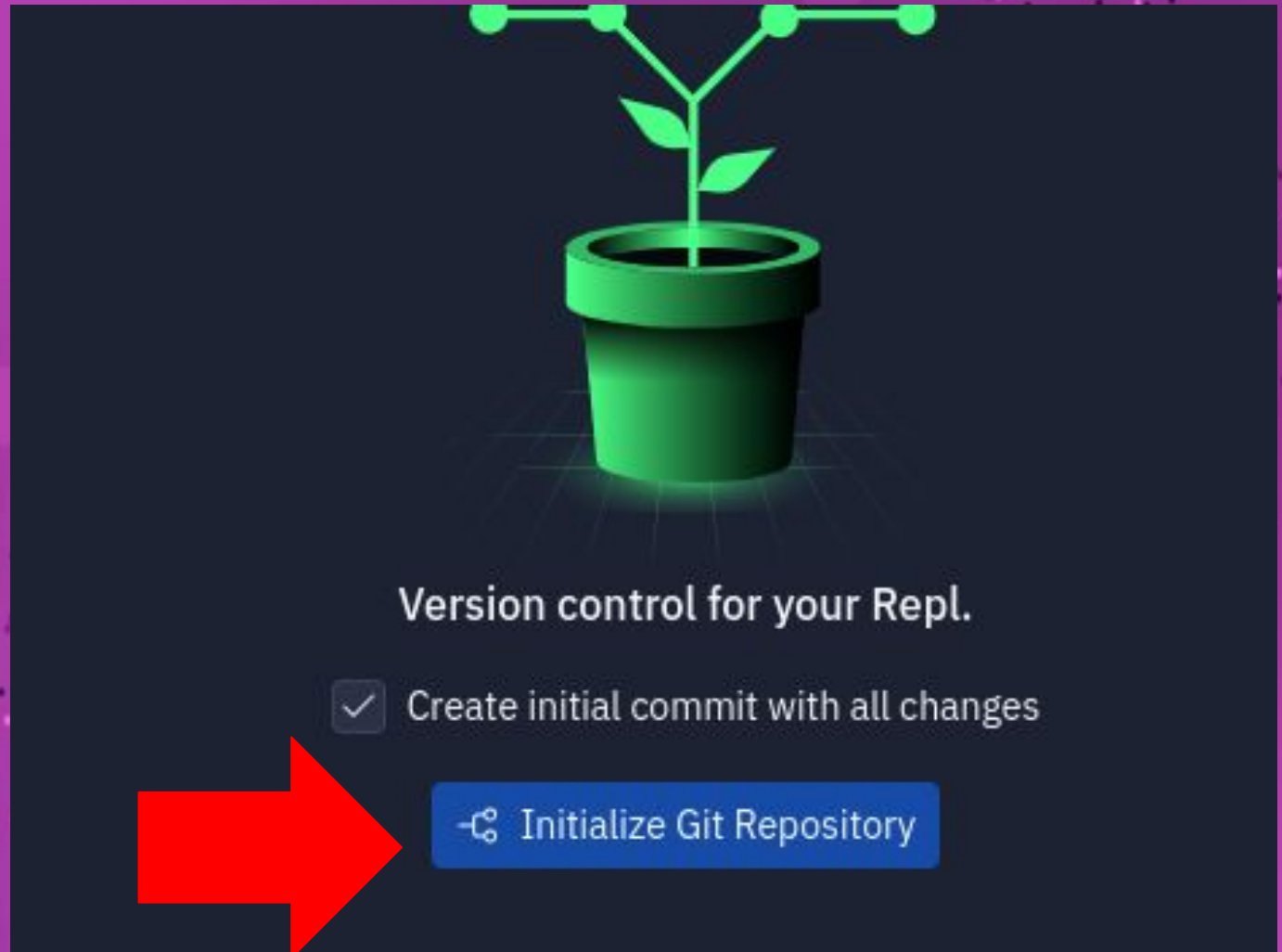
Funciona? Pra saber, só clicar em RUN, botão verde



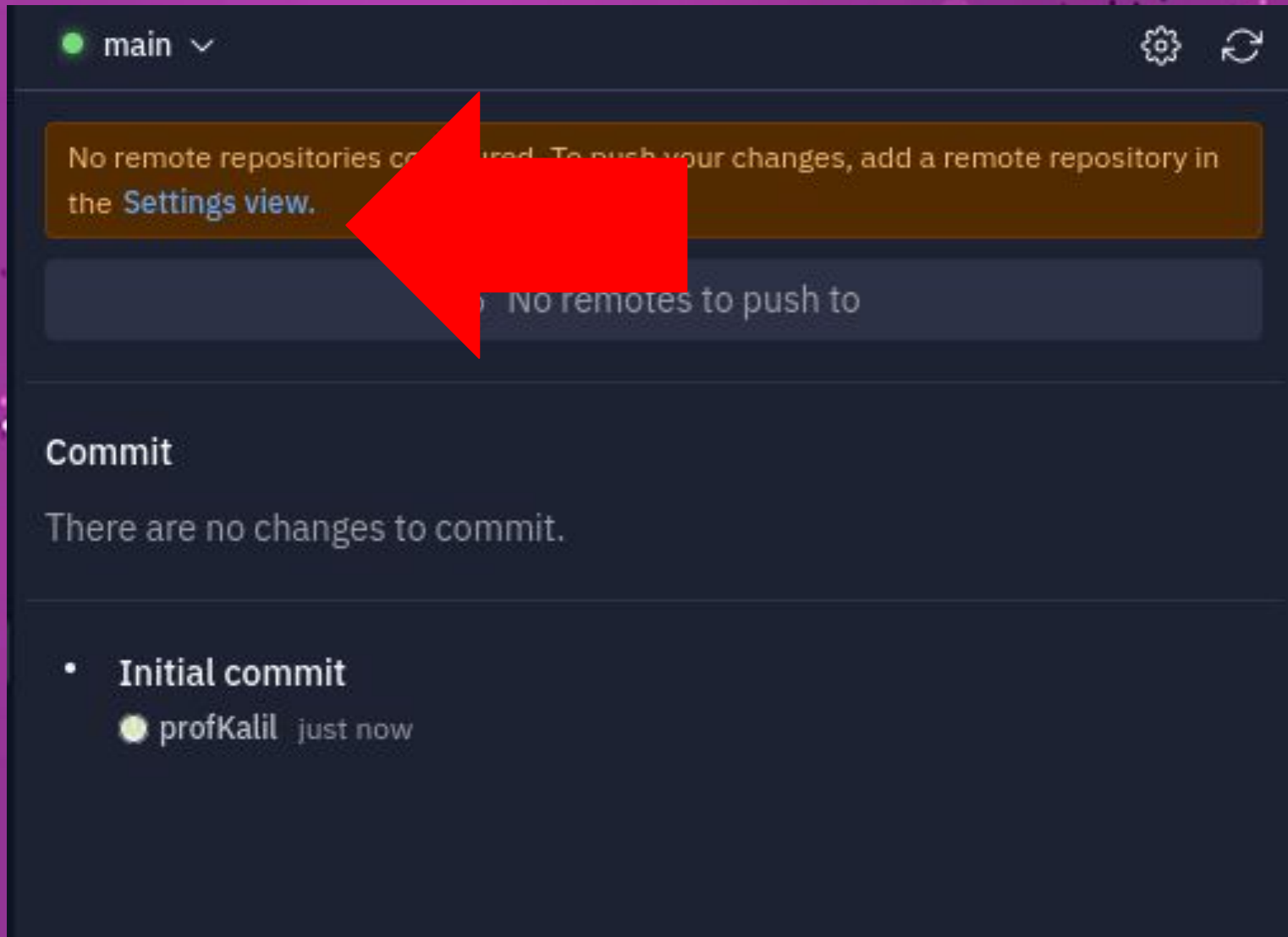
Agora só precisamos adicionar ao nosso github. No menu esquerdo, clicar em Tools e depois Git.




Você precisa inicializar o git no seu projeto e autorizar o repl.it a acessar o seu github.



Clicar em “Settings view” para as configurações...



Clique “Create...”. Espere. “< Settings” pra voltar!



< Settings

Remote

Enter Remote URL

GitHub

Repository name

somarumnumero

Repository description

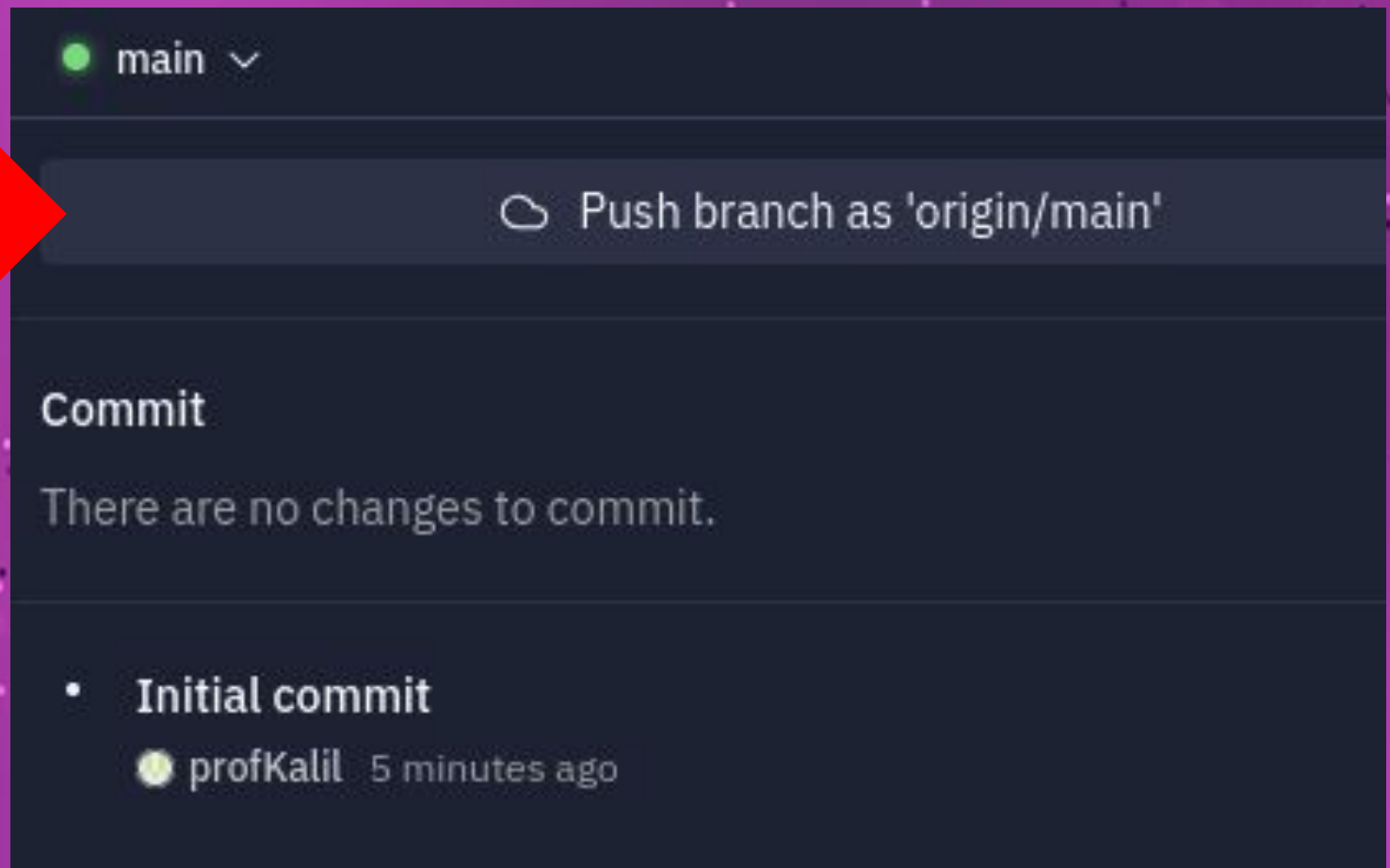
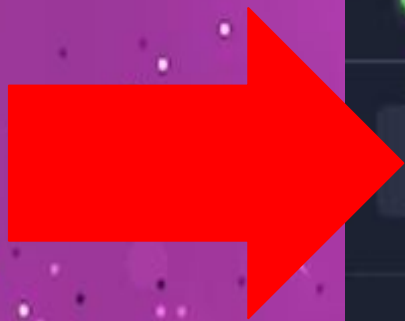
Repository for https://replit.com/@Kalilde/somarumnumero

Privacy

☒ Public ☐ Private

Create Repository on GitHub

Dê um “Push” (significa “empurrar”). Só clicar.



Alteramos o valor de *num2*. O git percebe que houve mudança e fica assim... Clique em “+” Você precisará escrever uma *Message*...(obrigatório) para dar commit

Commit


Message *

Commit: Ctrl Enter ▾

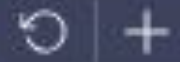
Summary of your commit...

Review Changes

1 changed file

 main.py

Modified



Committing will automatically stage your changes.

✓ Stage and commit all changes

Só falta fazer um Push (empurrar) e pronto! O repl.it retorna abaixo o commit com a sua Message e está tudo lá publicado no seu github. Simples assim.

Remote Updates profKalil/somarumnumero

origin/main • upstream last fetched 1 min ago Fetch

Nothing to pull or push

↓↑ Sync with Remote ↓ Pull ↑ Push

Commit

There are no changes to commit.

- **Meu primeiro commit**
● profKalil just now

The background is a vibrant purple gradient. It is decorated with numerous small white dots of varying sizes, resembling a starry night sky. Several larger, bright white stars with four-pointed flares are scattered across the image. At the top, there are wavy, horizontal lines in shades of blue and teal, creating a sense of depth and movement.

Vamos praticar!

<https://github.com/profkalil/logica-cedup>