# MAC 113 — Introdução à Ciência da Computação

## Aula 3

Nelson Lago

1°/2025



# Previously on MAC 113...

# Expressões em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: 47
  - Exemplo: $2 + 3$
  - Exemplo: "Oi galera!"
- **Expressões podem ser combinadas ou utilizadas como partes de outras expressões**
  - Exemplo: $2 + 3 + 7$
  - Exemplo: $\frac{2+3+7}{6}$

# Tipos

- **Existem *tipos de dados* diferentes em R**
  - Números inteiros, como 2 ou −437
    - » *Não são comumente usados em R*
    - » *Números entre ~−2 bilhões e ~2 bilhões*
  - Números (potencialmente) não-inteiros (“números de ponto flutuante”), como 2.0 ou 6.166666666666667
    - » *O tipo mais comum em R*
    - » *Às vezes o resultado de cálculos pode ser aproximado*
    - » 
```
options(digits=18)
cat(5^0.5 * 5^0.5, "\n")
```
```
5.00000000000000089
```
*Ok, números irracionais...*
```
cat(0.1 + 0.1 + 0.1, "\n")
```
```
0.300000000000000044
```
😱
  - Textos (“strings” ou “cadeias de caracteres”), como `"Oi galera!"`
- **R “sabe” quais operações podem ser realizadas com cada tipo**

# Nomes (variáveis)

- *Nomes* **permitem que pensemos mais sobre o problema a ser resolvido e menos sobre as idiossincrasias do computador**
- **Nomes em geral representam valores que *variam* (basicamente, alguma informação "real" que está em algum lugar na memória do computador)**
  - Como na matemática!
- **Por isso, chamamos esses nomes de "variáveis"**

# Nomes (variáveis)

x ⟵ 5 (atribuição)

Há um número finito de caracteres no teclado, então fazemos atribuição em R com "<-":

```
x <- 5
x <- "Olá, galera!"
x <- y
x <- x + 1
```

# and now for something completely different

# Formatação de strings

```
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat("Sensação térmica (°C):", round(sensação, 1), "\n")
```

# Formatação de strings

```
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat("Sensação térmica:", round(sensação, 1), "°C", "\n")
```

# Formatação de strings

```r
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat("Sensação térmica:", round(sensação, 1), "°C", "\n")
```

---

Velocidade do vento (Km/h): 10
Temperatura (°C): 25
Umidade (%): 50
Sensação térmica: 24.3 °C

# Formatação de strings

```r
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat("Sensação térmica:", round(sensação, 1), "°C", "\n")
```

---

Velocidade do vento (Km/h): 10
Temperatura (°C): 25
Umidade (%): 50
Sensação térmica: 24.3 °C

# Formatação de strings

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {sensação}°C"), "\n")
```

# Formatação de strings

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {sensação}°C"), "\n")
```

---

Velocidade do vento (Km/h): 10
Temperatura (°C): 25
Umidade (%): 50
Sensação térmica: 24.272094142423484°C

# Formatação de strings

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {sensação}°C"), "\n")
```

---

Velocidade do vento (Km/h): 10
Temperatura (°C): 25
Umidade (%): 50
Sensação térmica: 24.272094142423484°C

# Formatação de strings

```r
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {sensação}°C"), "\n")
```

---

```
Velocidade do vento (Km/h): 10
Temperatura (°C): 25
Umidade (%): 50
Sensação térmica: 24.272094142423484°C
```

# Formatação de strings

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {sensação}°C"), "\n")
```

---

```
Velocidade do vento (Km/h): 10
Temperatura (°C): 25
Umidade (%): 50
Sensação térmica: 24.272094142423484°C
```

# Formatação de strings

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {sensação}°C"), "\n")
```

---

```
Velocidade do vento (Km/h): 10
Temperatura (°C): 25
Umidade (%): 50
Sensação térmica: 24.272094142423484°C
```

# Formatação de strings

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

---

Velocidade do vento (Km/h): 10
Temperatura (°C): 25
Umidade (%): 50
Sensação térmica: 24.3°C

# Formatação de strings

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

---

```
Velocidade do vento (Km/h): 10
Temperatura (°C): 25
Umidade (%): 50
Sensação térmica: 24.3°C
```

- **O resultado de `glue()` é uma *expressão***

# Formatação de strings

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

---

```
Velocidade do vento (Km/h): 10
Temperatura (°C): 25
Umidade (%): 50
Sensação térmica: 24.3°C
```

- **O resultado de `glue()` é uma *expressão***
- **O valor da expressão é uma nova string**

# Exercícios

Dado um número $n$, diga seu equivalente em dúzias e unidades ("27 corresponde a 2 dúzias e 3 unidades").

# Exercícios

Dado um número $n$, diga seu equivalente em dúzias e unidades ("27 corresponde a 2 dúzias e 3 unidades").

```
n <- as.integer(readline("Digite um número natural: "))
```

# Exercícios

Dado um número $n$, diga seu equivalente em dúzias e unidades ("27 corresponde a 2 dúzias e 3 unidades").

```
n <- as.integer(readline("Digite um número natural: "))
dúzias <- n %/% 12
```

# Exercícios

Dado um número $n$, diga seu equivalente em dúzias e unidades ("27 corresponde a 2 dúzias e 3 unidades").

```
n <- as.integer(readline("Digite um número natural: "))
dúzias <- n %/% 12
unidades <- n %% 12
```

# Exercícios

Dado um número $n$, diga seu equivalente em dúzias e unidades ("27 corresponde a 2 dúzias e 3 unidades").

```
n <- as.integer(readline("Digite um número natural: "))
dúzias <- n %/% 12
unidades <- n %% 12
cat(n, "corresponde a", dúzias, "dúzias e", unidades, "unidades", "\n")
```

# Exercícios

Dado um número $n$, diga seu equivalente em dúzias e unidades ("27 corresponde a 2 dúzias e 3 unidades").

```
library(glue)
n <- as.integer(readline("Digite um número natural: "))
dúzias <- n %/% 12
unidades <- n %% 12
```

# Exercícios

Dado um número $n$, diga seu equivalente em dúzias e unidades (“27 corresponde a 2 dúzias e 3 unidades”).

```
library(glue)
n <- as.integer(readline("Digite um número natural: "))
dúzias <- n %/% 12
unidades <- n %% 12
cat(glue("{n} corresponde a {dúzias} dúzias e {unidades} unidades"), "\n")
```

# Exercícios

Dado um número $n$, diga seu equivalente em grosas, dúzias e unidades (“665 corresponde a 4 grosas, 7 dúzias e 5 unidades”).

```
library(glue)




cat(glue("{n} corresponde a {grosas} grosas, {dúzias} dúzias e {unidades} unidades"), "\n")
```

# Exercícios

Dado um número $n$, diga seu equivalente em grosas, dúzias e unidades (“665 corresponde a 4 grosas, 7 dúzias e 5 unidades”).

```
library(glue)
n <- as.integer(readline("Digite um número natural: "))




cat(glue("{n} corresponde a {grosas} grosas, {dúzias} dúzias e {unidades} unidades"), "\n")
```

# Exercícios

Dado um número $n$, diga seu equivalente em grosas, dúzias e unidades (“665 corresponde a 4 grosas, 7 dúzias e 5 unidades”).

```
library(glue)
n <- as.integer(readline("Digite um número natural: "))




unidades <- n %% 12
cat(glue("{n} corresponde a {grosas} grosas, {dúzias} dúzias e {unidades} unidades"), "\n")
```

# Exercícios

Dado um número $n$, diga seu equivalente em grosas, dúzias e unidades (“665 corresponde a 4 grosas, 7 dúzias e 5 unidades”).

```
library(glue)
n <- as.integer(readline("Digite um número natural: "))
grosas <- n %/% 144



unidades <- n %% 12
cat(glue("{n} corresponde a {grosas} grosas, {dúzias} dúzias e {unidades} unidades"), "\n")
```

# Exercícios

Dado um número $n$, diga seu equivalente em grosas, dúzias e unidades (“665 corresponde a 4 grosas, 7 dúzias e 5 unidades”).

```
library(glue)
n <- as.integer(readline("Digite um número natural: "))
grosas <- n %/% 144
soburô <- n %% 144

unidades <- n %% 12
cat(glue("{n} corresponde a {grosas} grosas, {dúzias} dúzias e {unidades} unidades"), "\n")
```

# Exercícios

Dado um número $n$, diga seu equivalente em grosas, dúzias e unidades (“665 corresponde a 4 grosas, 7 dúzias e 5 unidades”).

```
library(glue)
n <- as.integer(readline("Digite um número natural: "))
grosas <- n %/% 144
soburô <- n %% 144
dúzias <- soburô %/% 12
unidades <- n %% 12
cat(glue("{n} corresponde a {grosas} grosas, {dúzias} dúzias e {unidades} unidades"), "\n")
```

# Exercícios

Dado um número $n$, diga seu equivalente em grosas, dúzias e unidades (“665 corresponde a 4 grosas, 7 dúzias e 5 unidades”).

```
library(glue)
n <- as.integer(readline("Digite um número natural: "))
grosas <- n %/% 144
soburô <- n %% 144
dúzias <- soburô %/% 12
unidades <- soburô %% 12
cat(glue("{n} corresponde a {grosas} grosas, {dúzias} dúzias e {unidades} unidades"), "\n")
```

# Exercícios

Dado um número $n$, diga seu equivalente em grosas, dúzias e unidades (“665 corresponde a 4 grosas, 7 dúzias e 5 unidades”).

```
library(glue)
n <- as.integer(readline("Digite um número natural: "))
grosas <- n %/% 144
soburô <- n %% 144
dúzias <- soburô %/% 12
unidades <- n %% 12
cat(glue("{n} corresponde a {grosas} grosas, {dúzias} dúzias e {unidades} unidades"), "\n")
```

# Exercícios

Dado um número $n$, diga seu equivalente em grosas, dúzias e unidades (“665 corresponde a 4 grosas, 7 dúzias e 5 unidades”).

```
library(glue)
n <- as.integer(readline("Digite um número natural: "))
grosas <- n %/% 144

dúzias <- (n - grosas*144) %/% 12
unidades <- n %% 12
cat(glue("{n} corresponde a {grosas} grosas, {dúzias} dúzias e {unidades} unidades"), "\n")
```

# Nomes (funções)

- *Nomes* permitem que pensemos mais sobre o problema a ser resolvido e menos sobre as idiossincrasias do computador

# Nomes (funções)

- *Nomes* permitem que pensemos mais sobre o problema a ser resolvido e menos sobre as idiossincrasias do computador
- Variáveis são **expressões** (têm um valor)

# Nomes (funções)

- *Nomes* permitem que pensemos mais sobre o problema a ser resolvido e menos sobre as idiossincrasias do computador
- Variáveis são **expressões** (têm um valor)
- Nomes também podem ser usados para representar "ações" (**como calcular** o valor de uma expressão)

# Nomes (funções)

- ***Nomes* permitem que pensemos mais sobre o problema a ser resolvido e menos sobre as idiossincrasias do computador**
- **Variáveis são expressões (têm um valor)**
- **Nomes também podem ser usados para representar "ações" (como calcular o valor de uma expressão)**
  - Por exemplo, `sqrt()` é o nome que usamos para representar o cálculo da raiz quadrada

# Nomes (funções)

- ***Nomes* permitem que pensemos mais sobre o problema a ser resolvido e menos sobre as idiossincrasias do computador**
- **Variáveis são expressões (têm um valor)**
- **Nomes também podem ser usados para representar "ações" (como calcular o valor de uma expressão)**
  - Por exemplo, `sqrt()` é o nome que usamos para representar o cálculo da raiz quadrada
- **Nomes desse tipo recebem um valor, realizam algum processamento e devolvem outro valor correspondente**

# Nomes (funções)

- ***Nomes* permitem que pensemos mais sobre o problema a ser resolvido e menos sobre as idiossincrasias do computador**
- **Variáveis são expressões (têm um valor)**
- **Nomes também podem ser usados para representar "ações" (como calcular o valor de uma expressão)**
  - Por exemplo, `sqrt()` é o nome que usamos para representar o cálculo da raiz quadrada
- **Nomes desse tipo recebem um valor, realizam algum processamento e devolvem outro valor correspondente**
  - Como as funções da matemática

# Nomes (funções)

- ***Nomes* permitem que pensemos mais sobre o problema a ser resolvido e menos sobre as idiossincrasias do computador**
- **Variáveis são expressões (têm um valor)**
- **Nomes também podem ser usados para representar "ações" (como calcular o valor de uma expressão)**
  - Por exemplo, `sqrt()` é o nome que usamos para representar o cálculo da raiz quadrada
- **Nomes desse tipo recebem um valor, realizam algum processamento e devolvem outro valor correspondente**
  - Como as funções da matemática
- **E, por isso, também são chamados de funções**

# Nomes (funções)

```r
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

# Nomes (funções)

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

- **Aqui, sensação representa o *resultado* da expressão**

# Nomes (funções)

```r
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

- **Aqui, sensação representa o *resultado* da expressão**
- **Mas podemos representar também o *cálculo* da expressão**

# Nomes (funções)

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

- **Aqui, sensação representa o *resultado* da expressão**
- **Mas podemos representar também o *cálculo* da expressão**

```
library(glue)
```

# Nomes (funções)

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

- **Aqui, sensação representa o *resultado* da expressão**
- **Mas podemos representar também o *cálculo* da expressão**

```
library(glue)
            function(v, t, u)
```

# Nomes (funções)

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

- **Aqui, sensação representa o *resultado* da expressão**
- **Mas podemos representar também o *cálculo* da expressão**

```
library(glue)
            function(v, t, u) {

}
```

# Nomes (funções)

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

- **Aqui, sensação representa o *resultado* da expressão**
- **Mas podemos representar também o *cálculo* da expressão**

```
library(glue)
            function(v, t, u) {
    return(                                                    )
}
```

# Nomes (funções)

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

- **Aqui, sensação representa o *resultado* da expressão**
- **Mas podemos representar também o *cálculo* da expressão**

```
library(glue)
            function(v, t, u) {
    return(round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1))
}
```

# Nomes (funções)

```r
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

- **Aqui, sensação representa o *resultado* da expressão**
- **Mas podemos representar também o *cálculo* da expressão**

```r
library(glue)
sensação <- function(v, t, u) {
    return(round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1))
}
```

# Nomes (funções)

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

- **Aqui, sensação representa o *resultado* da expressão**
- **Mas podemos representar também o *cálculo* da expressão**

```
library(glue)
sensação <- function(v, t, u) {
    return(round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
```

# Nomes (funções)

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

- **Aqui, sensação representa o *resultado* da expressão**
- **Mas podemos representar também o *cálculo* da expressão**

```
library(glue)
sensação <- function(v, t, u) {
    return(round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {
    return(round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {

    return(round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {
    resultado = t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
    return(      resultado    )
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {
    resultado = t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
    return(round(resultado, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {
    resultado = t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
    return(round(resultado, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

- **A função define como realizar uma operação**

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {
    resultado = t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
    return(round(resultado, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

- A função define **como** realizar uma operação
- A **chamada** à função é uma expressão (tem um valor)

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {
    resultado = t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
    return(round(resultado, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

- A função define **como** realizar uma operação
- A **chamada** à função é uma expressão (tem um valor)
- O começo e o fim da função são identificados por { e }

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {
⟷ resultado = t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
⟷ return(round(resultado, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

- A função define **como** realizar uma operação
- A **chamada** à função é uma expressão (tem um valor)
- O começo e o fim da função são identificados por { e }

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {
⟷ resultado = t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
⟷ return(round(resultado, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

- A função define **como** realizar uma operação
- A **chamada** à função é uma expressão (tem um valor)
- O começo e o fim da função são identificados por { e }
- A indentação em R é apenas uma **convenção**

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {
⟷ resultado = t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
⟷ return(round(resultado, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

- **A função define como realizar uma operação**
- **A chamada à função é uma expressão (tem um valor)**
- **O começo e o fim da função são identificados por { e }**
- **A indentação em R é apenas uma convenção**
  - Mas é muito importante e comum a todas as linguagens de programação

# Nomes (funções)

```
library(glue)
sensação <- function(v, t, u) {
⟷ resultado = t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
⟷ return(round(resultado, 1))
}
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

- **A função define como realizar uma operação**
- **A chamada à função é uma expressão (tem um valor)**
- **O começo e o fim da função são identificados por { e }**
- **A indentação em R é apenas uma convenção**
  - Mas é muito importante e comum a todas as linguagens de programação
    - » *Em python não é apenas convenção, é obrigatório*

# Função `main()`

- Vamos nos acostumar a fazer nossos programas dentro de uma função chamada `main()`

# Função `main()`

- **Vamos nos acostumar a fazer nossos programas dentro de uma função chamada `main()`**

```
library(glue)
sensação <- function(v, t, u) {
    resultado = round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1)
    return(resultado)
}

    v <- as.numeric(readline("Velocidade do vento (Km/h): "))
    t <- as.numeric(readline("Temperatura (°C): "))
    u <- as.numeric(readline("Umidade relativa (%): ")) / 100
    cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
```

# Função `main()`

- **Vamos nos acostumar a fazer nossos programas dentro de uma função chamada `main()`**

```
library(glue)
sensação <- function(v, t, u) {
    resultado = round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1)
    return(resultado)
}
                    {
    v <- as.numeric(readline("Velocidade do vento (Km/h): "))
    t <- as.numeric(readline("Temperatura (°C): "))
    u <- as.numeric(readline("Umidade relativa (%): ")) / 100
    cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
}
```

# Função `main()`

- **Vamos nos acostumar a fazer nossos programas dentro de uma função chamada `main()`**

```
library(glue)
sensação <- function(v, t, u) {
    resultado = round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1)
    return(resultado)
}
                () {
    v <- as.numeric(readline("Velocidade do vento (Km/h): "))
    t <- as.numeric(readline("Temperatura (°C): "))
    u <- as.numeric(readline("Umidade relativa (%): ")) / 100
    cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
}
```

# Função `main()`

- **Vamos nos acostumar a fazer nossos programas dentro de uma função chamada `main()`**

```
library(glue)
sensação <- function(v, t, u) {
    resultado = round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1)
    return(resultado)
}
        function() {
    v <- as.numeric(readline("Velocidade do vento (Km/h): "))
    t <- as.numeric(readline("Temperatura (°C): "))
    u <- as.numeric(readline("Umidade relativa (%): ")) / 100
    cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
}
```

# Função `main()`

- Vamos nos acostumar a fazer nossos programas dentro de uma função chamada `main()`

```
library(glue)
sensação <- function(v, t, u) {
    resultado = round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1)
    return(resultado)
}
main <- function() {
    v <- as.numeric(readline("Velocidade do vento (Km/h): "))
    t <- as.numeric(readline("Temperatura (°C): "))
    u <- as.numeric(readline("Umidade relativa (%): ")) / 100
    cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
}
```

# Função `main()`

- Vamos nos acostumar a fazer nossos programas dentro de uma função chamada `main()`

```
library(glue)
sensação <- function(v, t, u) {
    resultado = round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1)
    return(resultado)
}
main <- function() {
    v <- as.numeric(readline("Velocidade do vento (Km/h): "))
    t <- as.numeric(readline("Temperatura (°C): "))
    u <- as.numeric(readline("Umidade relativa (%): ")) / 100
    cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
}
main()
```

# Função `main()`

- Vamos nos acostumar a fazer nossos programas dentro de uma função chamada `main()`

```
library(glue)
sensação <- function(v, t, u) {
    resultado = round(t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4, 1)
    return(resultado)
}
main <- function() {
    v <- as.numeric(readline("Velocidade do vento (Km/h): "))
    t <- as.numeric(readline("Temperatura (°C): "))
    u <- as.numeric(readline("Umidade relativa (%): ")) / 100
    cat(glue("Sensação térmica: {sensação(v, t, u)}°C"), "\n")
}
main()
```

- Isso também é apenas uma convenção, mas vai facilitar a vida mais à frente 😉

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```r
f <- as.numeric(readline("Qual a temperatura? "))
c <- 5 * (f - 32) / 9
cat(f, "graus Fahrenheit correspondem a", round(c, 1), "graus Celsius", "\n")
```

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```
f <- as.numeric(readline("Qual a temperatura? "))
c <- 5 * (f - 32) / 9
cat(f, "graus Fahrenheit correspondem a", round(c, 1), "graus Celsius", "\n")
```

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```
main <- function() {
    f <- as.numeric(readline("Qual a temperatura? "))
    c <- 5 * (f - 32) / 9
    cat(f, "graus Fahrenheit correspondem a", round(c, 1), "graus Celsius", "\n")
}
main()
```

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```
main <- function() {
    f <- as.numeric(readline("Qual a temperatura? "))

    cat(f, "graus Fahrenheit correspondem a", round(c, 1), "graus Celsius", "\n")
}
main()
```

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```
main <- function() {
    f <- as.numeric(readline("Qual a temperatura? "))
    cat(f, "graus Fahrenheit correspondem a", celsius(f), "graus Celsius", "\n")
}
main()
```

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```
          function(f)


main <- function() {
    f <- as.numeric(readline("Qual a temperatura? "))
    cat(f, "graus Fahrenheit correspondem a", celsius(f), "graus Celsius", "\n")
}
main()
```

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```
          function(f) {

}
main <- function() {
    f <- as.numeric(readline("Qual a temperatura? "))
    cat(f, "graus Fahrenheit correspondem a", celsius(f), "graus Celsius", "\n")
}
main()
```

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```
           function(f) {
    return(                          )
}
main <- function() {
    f <- as.numeric(readline("Qual a temperatura? "))
    cat(f, "graus Fahrenheit correspondem a", celsius(f), "graus Celsius", "\n")
}
main()
```

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```
           function(f) {
    return(      5 * (f - 32) / 9    )
}
main <- function() {
    f <- as.numeric(readline("Qual a temperatura? "))
    cat(f, "graus Fahrenheit correspondem a", celsius(f), "graus Celsius", "\n")
}
main()
```

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```
            function(f) {
    return(round(5 * (f - 32) / 9, 1))
}
main <- function() {
    f <- as.numeric(readline("Qual a temperatura? "))
    cat(f, "graus Fahrenheit correspondem a", celsius(f), "graus Celsius", "\n")
}
main()
```

# Exercício

Modifique o conversor de temperaturas abaixo para usar uma função

```
celsius <- function(f) {
    return(round(5 * (f - 32) / 9, 1))
}
main <- function() {
    f <- as.numeric(readline("Qual a temperatura? "))
    cat(f, "graus Fahrenheit correspondem a", celsius(f), "graus Celsius", "\n")
}
main()
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
a <- 3
b <- 4
c <- 5
s <- (a + b + c) / 2
resultado = sqrt(s * (s-a) * (s-b) * (s-c))
cat("A área do triângulo dado é", resultado, "\n")
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
a <- as.integer(readline("Digite o primeiro lado: ")
b <- as.integer(readline("Digite o segundo lado: ")
c <- as.integer(readline("Digite o terceiro lado: ")
s <- (a + b + c) / 2
resultado = sqrt(s * (s-a) * (s-b) * (s-c))
cat("A área do triângulo dado é", resultado, "\n")
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
a <- as.integer(readline("Digite o primeiro lado: ")
b <- as.integer(readline("Digite o segundo lado: ")
c <- as.integer(readline("Digite o terceiro lado: ")
s <- (a + b + c) / 2
resultado = sqrt(s * (s-a) * (s-b) * (s-c))
cat("A área do triângulo dado é", resultado, "\n")
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
main <- function() {
    a <- as.integer(readline("Digite o primeiro lado: ")
    b <- as.integer(readline("Digite o segundo lado: ")
    c <- as.integer(readline("Digite o terceiro lado: ")
    s <- (a + b + c) / 2
    resultado = sqrt(s * (s-a) * (s-b) * (s-c))
    cat("A área do triângulo dado é", resultado, "\n")
}
main()
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
main <- function() {
    a <- as.integer(readline("Digite o primeiro lado: ")
    b <- as.integer(readline("Digite o segundo lado: ")
    c <- as.integer(readline("Digite o terceiro lado: ")


    cat("A área do triângulo dado é", area(a, b, c), "\n")
}
main()
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
main <- function() {
    a <- as.integer(readline("Digite o primeiro lado: ")
    b <- as.integer(readline("Digite o segundo lado: ")
    c <- as.integer(readline("Digite o terceiro lado: ")
    cat("A área do triângulo dado é", area(a, b, c), "\n")
}
main()
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
area <- function(a, b, c)



main <- function() {
    a <- as.integer(readline("Digite o primeiro lado: ")
    b <- as.integer(readline("Digite o segundo lado: ")
    c <- as.integer(readline("Digite o terceiro lado: ")
    cat("A área do triângulo dado é", area(a, b, c), "\n")
}
main()
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
area <- function(a, b, c) {


}
main <- function() {
    a <- as.integer(readline("Digite o primeiro lado: ")
    b <- as.integer(readline("Digite o segundo lado: ")
    c <- as.integer(readline("Digite o terceiro lado: ")
    cat("A área do triângulo dado é", area(a, b, c), "\n")
}
main()
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
area <- function(a, b, c) {
    s <- (a + b + c) / 2

}
main <- function() {
    a <- as.integer(readline("Digite o primeiro lado: ")
    b <- as.integer(readline("Digite o segundo lado: ")
    c <- as.integer(readline("Digite o terceiro lado: ")
    cat("A área do triângulo dado é", area(a, b, c), "\n")
}
main()
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
area <- function(a, b, c) {
    s <- (a + b + c) / 2
    return(                                  )
}
main <- function() {
    a <- as.integer(readline("Digite o primeiro lado: ")
    b <- as.integer(readline("Digite o segundo lado: ")
    c <- as.integer(readline("Digite o terceiro lado: ")
    cat("A área do triângulo dado é", area(a, b, c), "\n")
}
main()
```

# Exercício – área do triângulo

A área de um triângulo de lados $a, b, c$ é dada por $\sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$, onde $s = \frac{a+b+c}{2}$

Modifique o programa abaixo para obter os valores de $a, b, c$ do usuário e calcular a área usando uma função

```
area <- function(a, b, c) {
    s <- (a + b + c) / 2
    return(sqrt(s * (s-a) * (s-b) * (s-c)))
}
main <- function() {
    a <- as.integer(readline("Digite o primeiro lado: ")
    b <- as.integer(readline("Digite o segundo lado: ")
    c <- as.integer(readline("Digite o terceiro lado: ")
    cat("A área do triângulo dado é", area(a, b, c), "\n")
}
main()
```

# Expressões em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)

# Expressões em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $2 > 3$

# Expressões em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $2 > 3$
- **AAAAAHHHH!!!!!!!**

# Expressões em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $2 > 3$
- **AAAAAHHHH!!!!!!!**
- **Qual o resultado dessa heresia?**

# Expressões em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $2 > 3$
- **AAAAAHHHH!!!!!!!**
- **Qual o resultado dessa heresia?**

```
cat(2 > 3, "\n")
```

# Expressões em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $2 > 3$
- **AAAAAHHHH!!!!!!!**
- **Qual o resultado dessa heresia?**

```
cat(2 > 3, "\n")
```

```
FALSE
```

# Expressões em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $2 > 3$
- **AAAAAHHHH!!!!!!!**
- **Qual o resultado dessa heresia?**

```
cat(2 > 3, "\n")
```

```
FALSE
```

**Ufa!**

# Tipos booleanos

- O que exatamente são `TRUE` e `FALSE`?

# Tipos booleanos

- O que exatamente são `TRUE` e `FALSE`?
- São um outro *tipo* de dado (como inteiros, *strings* e números de ponto flutuante)

# Tipos booleanos

- O que exatamente são `TRUE` e `FALSE`?
- São um outro *tipo* de dado (como inteiros, *strings* e números de ponto flutuante)
- Esse tipo se chama *booleano* (em homenagem a George Boole)

# Tipos booleanos

- **O que exatamente são `TRUE` e `FALSE`?**
- **São um outro *tipo* de dado (como inteiros, *strings* e números de ponto flutuante)**
- **Esse tipo se chama *booleano* (em homenagem a George Boole)**
  - Em R, o nome usado é *logical*

# Tipos booleanos

- **O que exatamente são `TRUE` e `FALSE`?**
- **São um outro *tipo* de dado (como inteiros, *strings* e números de ponto flutuante)**
- **Esse tipo se chama *booleano* (em homenagem a George Boole)**
  - Em R, o nome usado é *logical*
- **Tipos booleanos só podem ter dois valores: `TRUE` e `FALSE`**

# Tipos

```r
cat(class(2L), "\n")
cat(class(FALSE), "\n")
cat(class(2.0), "\n")
cat(class("Olá"), "\n")
```

# Tipos

```
cat(class(2L), "\n")
cat(class(FALSE), "\n")
cat(class(2.0), "\n")
cat(class("Olá"), "\n")
```

---

integer

# Tipos

```r
cat(class(2L), "\n")
cat(class(FALSE), "\n")
cat(class(2.0), "\n")
cat(class("Olá"), "\n")
```

---

integer
logical

# Tipos

```r
cat(class(2L), "\n")
cat(class(FALSE), "\n")
cat(class(2.0), "\n")
cat(class("Olá"), "\n")
```

```
integer
logical
numeric
```

# Tipos

```r
cat(class(2L), "\n")
cat(class(FALSE), "\n")
cat(class(2.0), "\n")
cat(class("Olá"), "\n")
```

---

```
integer
logical
numeric
character
```

# Operadores relacionais

| operador | descrição |
|---|---|
| == | igualdade |
| != | desigualdade |
| > | maior |
| >= | maior ou igual ($\geq$) |
| < | menor |
| <= | menor ou igual ($\leq$) |

*Operadores relacionais em R*

# Operadores relacionais

| operador | descrição |
|---|---|
| == | igualdade |
| != | desigualdade |
| > | maior |
| >= | maior ou igual ($\geq$) |
| < | menor |
| <= | menor ou igual ($\leq$) |

*Operadores relacionais em R*

# Programando em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)

# Programando em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: 47

# Programando em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: 47
  - Exemplo: $2 + 3$

# Programando em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $47$
  - Exemplo: $2 + 3$
  - Exemplo: $2 > 3$

# Programando em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $47$
  - Exemplo: $2 + 3$
  - Exemplo: $2 > 3$
- **Expressões podem ser combinadas ou utilizadas como partes de outras expressões**

# Programando em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $47$
  - Exemplo: $2 + 3$
  - Exemplo: $2 > 3$
- **Expressões podem ser combinadas ou utilizadas como partes de outras expressões**
  - Exemplo: $\frac{2+3+7}{6}$

# Programando em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $47$
  - Exemplo: $2 + 3$
  - Exemplo: $2 > 3$
- **Expressões podem ser combinadas ou utilizadas como partes de outras expressões**
  - Exemplo: $\frac{2+3+7}{6}$
  - Exemplo: $2 + 3 + 7 < 4$

# Programando em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $47$
  - Exemplo: $2 + 3$
  - Exemplo: $2 > 3$
- **Expressões podem ser combinadas ou utilizadas como partes de outras expressões**
  - Exemplo: $\frac{2+3+7}{6}$
  - Exemplo: $2 + 3 + 7 < 4$
  - Exemplo: $2 + 2 == 2 * 2$

# Programando em R

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $47$
  - Exemplo: $2 + 3$
  - Exemplo: $2 > 3$
- **Expressões podem ser combinadas ou utilizadas como partes de outras expressões**
  - Exemplo: $\frac{2+3+7}{6}$
  - Exemplo: $2 + 3 + 7 < 4$
  - Exemplo: $2 + 2 == 2 * 2$

# Operadores relacionais

```
cat(2 + 3 + 7 < 4, 2 + 2 == 2 * 2, "\n")
```

# Operadores relacionais

```
cat(2 + 3 + 7 < 4, 2 + 2 == 2 * 2, "\n")
```

```
FALSE TRUE
```

# Operadores relacionais

```
cat(2 + 3 + 7 < 4, 2 + 2 == 2 * 2, "\n")
```

```
FALSE TRUE
```

**Os operandos são inteiros, mas o resultado da expressão é um booleano**

# Expressões lógicas

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)

# Expressões lógicas

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: 47

# Expressões lógicas

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: 47
  - Exemplo: $2 + 3$

# Expressões lógicas

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $47$
  - Exemplo: $2 + 3$
  - Exemplo: $2 > 3$

# Expressões lógicas

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $47$
  - Exemplo: $2 + 3$
  - Exemplo: $2 > 3$
- **Expressões podem ser combinadas ou utilizadas como partes de outras expressões**

# Expressões lógicas

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: $47$
  - Exemplo: $2 + 3$
  - Exemplo: $2 > 3$
- **Expressões podem ser combinadas ou utilizadas como partes de outras expressões**
  - Exemplo: $2 + 3 + 7 < 4$

# Expressões lógicas

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: `47`
  - Exemplo: `2 + 3`
  - Exemplo: `2 > 3`
- **Expressões podem ser combinadas ou utilizadas como partes de outras expressões**
  - Exemplo: `2 + 3 + 7 < 4`
  - Exemplo: `2 > 3 && 5 > 4`

# Expressões lógicas

- **A maioria das coisas em R são *expressões***
  - (expressões são coisas que têm um *valor*)
  - Exemplo: 47
  - Exemplo: 2 + 3
  - Exemplo: 2 > 3
- **Expressões podem ser combinadas ou utilizadas como partes de outras expressões**
  - Exemplo: 2 + 3 + 7 < 4
  - Exemplo: 2 > 3 && 5 > 4

# Expressões lógicas

```
cat(2 > 3, "\n")
cat(5 > 4, "\n")
cat(2 > 3 && 5 > 4, "\n")
cat(2 > 3 || 5 > 4, "\n")
```

# Expressões lógicas

```
cat(2 > 3, "\n")
cat(5 > 4, "\n")
cat(2 > 3 && 5 > 4, "\n")
cat(2 > 3 || 5 > 4, "\n")
```

```
FALSE
```

# Expressões lógicas

```
cat(2 > 3, "\n")
cat(5 > 4, "\n")
cat(2 > 3 && 5 > 4, "\n")
cat(2 > 3 || 5 > 4, "\n")
```

```
FALSE
TRUE
```

# Expressões lógicas

```r
cat(2 > 3, "\n")
cat(5 > 4, "\n")
cat(2 > 3 && 5 > 4, "\n")
cat(2 > 3 || 5 > 4, "\n")
```

```
FALSE
TRUE
FALSE
```

# Expressões lógicas

```
cat(2 > 3, "\n")
cat(5 > 4, "\n")
cat(2 > 3 && 5 > 4, "\n")
cat(2 > 3 || 5 > 4, "\n")
```

```
FALSE
TRUE
FALSE
TRUE
```

# Precedência de operadores

| operador | descrição |
|---|---|
| ! | negação lógica (“inverte o sinal”) |
| && | E lógico (só é verdadeiro se ambos os operandos são verdadeiros) |
| \|\| | OU lógico (só é verdadeiro se ao menos um dos operandos é verdadeiro) |

*Precedência dos operadores lógicos em R (da maior para a menor)*

# Operadores lógicos

A && B

| | A = **TRUE** | A = **FALSE** |
|---|---|---|
| B = **TRUE** | **TRUE** | **FALSE** |
| B = **FALSE** | **FALSE** | **FALSE** |

*Tabela verdade do operador* &&

# Operadores lógicos

A || B

| | A = **TRUE** | A = **FALSE** |
|---|---|---|
| B = **TRUE** | **TRUE** | **TRUE** |
| B = **FALSE** | **TRUE** | **FALSE** |

*Tabela verdade do operador* ||

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

```
cat(                                                    , "\n")
```

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

```
cat(temp > 25                                                    , "\n")
```

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

```
cat(temp > 25 && saldo >= 100                                    , "\n")
```

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

```
cat(temp > 25 && saldo >= 100 &&                                    , "\n")
```

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

```
cat(temp > 25 && saldo >= 100 &&  nota == "A" || nota == "B" , "\n")
```

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

```
cat(temp > 25 && saldo >= 100 && (nota == "A" || nota == "B"), "\n")
```

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

```
temp <- 29L
saldo <- 100.39
nota <- "B"
cat(temp > 25 && saldo >= 100 && (nota == "A" || nota == "B"), "\n")
```

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

```
temp <- 29L
saldo <- 100.39
nota <- "B"
cat(temp > 25 && saldo >= 100 && (nota == "A" || nota == "B"), "\n")
```

```
TRUE
```

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

```
temp <- 29L
saldo <- 100.39
nota <- "B"
cat(temp > 25 && saldo >= 100 && (nota == "A" || nota == "B"), "\n")
```

integer

```
TRUE
```

# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**

```
temp <- 29L
saldo <- 100.39
nota <- "B"
cat(temp > 25 && saldo >= 100 && (nota == "A" || nota == "B"), "\n")
```

integer

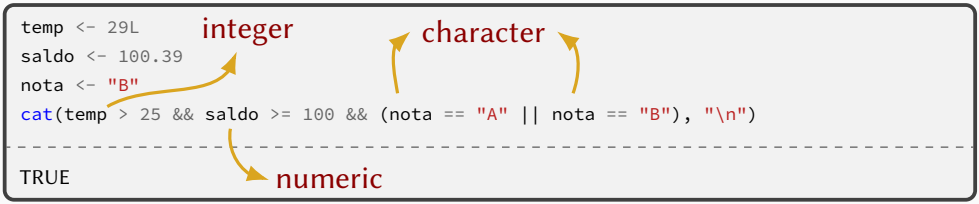
numeric

```
TRUE
```

# Expressões lógicas e tipos

Só vou à praia se:

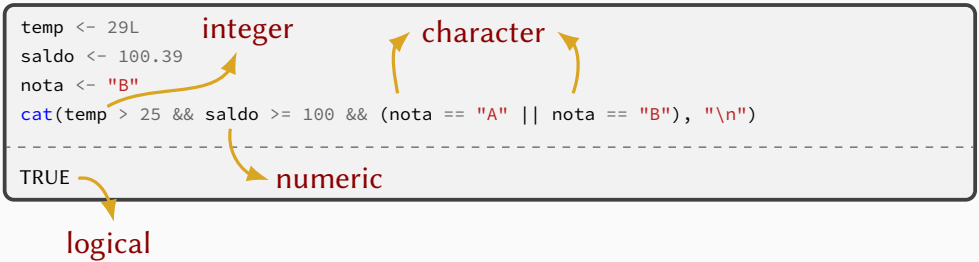
- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**



# Expressões lógicas e tipos

Só vou à praia se:

- **A temperatura no fim-de-semana for maior que 25°C**
- **Eu tiver pelo menos R$100**
- **A minha nota na prova for A ou B**



# E para que serve isso?

# Execução condicional

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
```

# Execução condicional

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
if (sensação < 18)
```

# Execução condicional

```
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
if (sensação < 18) {

}
```

# Execução condicional

```r
library(glue)
v <- as.numeric(readline("Velocidade do vento (Km/h): "))
t <- as.numeric(readline("Temperatura (°C): "))
u <- as.numeric(readline("Umidade relativa (%): ")) / 100
sensação <- t + 2.015 * u * exp(17.27*t/(237.7+t)) - 0.194*v - 4
cat(glue("Sensação térmica: {round(sensação, 1)}°C"), "\n")
if (sensação < 18) {
    cat("Leva um casaquinho!", "\n")
}
```