**Para saber mais: função ifelse()**

No universo da programação em R, a função ifelse() desempenha um papel crucial ao facilitar a tomada de decisões em um contexto vetorizado. Ela oferece uma alternativa eficiente e concisa às construções condicionais convencionais if() e else().

A função ifelse() permite aplicar condições a cada elemento de um vetor ou matriz de maneira eficiente. Ela segue a sintaxe:

**ifelse(test, yes, no)**

* test: Uma expressão lógica que avalia se uma condição é verdadeira ou falsa.
* yes: Valor a ser atribuído se a condição for verdadeira.
* no: Valor a ser atribuído se a condição for falsa.

**Exemplos Práticos**

*Exemplo 1: Aplicando Condição a um Vetor*

Suponha que você tenha um vetor de idades e deseje categorizá-las como "Adulto" se forem maiores ou iguais a 18, e "Jovem" caso contrário.

idades <- c(25, 16, 22, 30, 14)

categorias <- ifelse(idades >= 18, "Adulto", "Jovem")

print(categorias)

Neste exemplo, a função ifelse() avalia a condição para cada elemento de idades, atribuindo "Adulto" se a idade for maior ou igual a 18 e "Jovem" caso contrário.

*Exemplo 2: Aplicando Condição a uma Matriz*

Imagine uma matriz de notas de alunos, onde você deseja atribuir "Aprovado" se a nota for maior ou igual a 60 e "Reprovado" caso contrário.

notas <- matrix(c(75, 45, 80, 55, 90, 65), ncol = 2)

resultados <- ifelse(notas >= 60, "Aprovado", "Reprovado")

print(resultados)

### Comparação com if() e else()

Enquanto ifelse() é vetorizada e adequada para aplicação em vetores e matrizes, if() e else() são construções condicionais tradicionais que atuam em um único valor por vez. A principal diferença reside na capacidade de ifelse() lidar eficientemente com operações vetorizadas, resultando em código mais compacto e legível.

Exemplo com if() e else():

idade <- 20

**if** (idade >= 18) {

categoria <- "Adulto"

} **else** {

categoria <- "Jovem"

}

Exemplo equivalente com ifelse():

idade <- 20

categoria <- ifelse(idade >= 18, "Adulto", "Jovem")

A função ifelse() destaca-se quando a vetorização é essencial, proporcionando uma abordagem mais eficiente para a tomada de decisões em conjuntos de dados. No entanto, a escolha entre ifelse() e as construções condicionais tradicionais depende do contexto e dos requisitos específicos de cada situação.

**Para saber mais: operadores do R**

Em R, os operadores são elementos fundamentais que possibilitam a realização de diversas operações em dados. Eles são categorizados em operadores aritméticos, lógicos e de comparação, cada um desempenhando um papel específico nas análises e manipulações de dados.

**Operadores Lógicos**

Os operadores lógicos, como &, |, &&, ||, e !, são utilizados para combinar condições lógicas. Eles ajudam a formular expressões complexas e a realizar avaliações booleanas, facilitando a tomada de decisões em estruturas condicionais.

Podemos conferir alguns exemplos de operadores lógicos abaixo:

* & (E lógico): Retorna TRUE se ambas as condições forem verdadeiras.
* && (E lógico curto-circuito): Semelhante a &, mas avalia apenas a primeira condição se a segunda não for necessária.
* | (OU lógico): Retorna TRUE se pelo menos uma das condições for verdadeira.
* || (OU lógico curto-circuito): Semelhante a |, mas avalia apenas a primeira condição se a segunda não for necessária.
* ! (NÃO lógico): Inverte o valor lógico, de TRUE para FALSE e vice-versa.

**Operadores de Comparação**

Operadores de comparação são usados para comparar valores e gerar expressões booleanas. Esses operadores são fundamentais para avaliar relações entre variáveis e tomar decisões com base nessas comparações.

A seguir estão alguns exemplos de operadores de comparação:

* == (igual a)
* != (diferente de)
* < (menor que)
* > (maior que)
* <= (menor ou igual a)
* >= (maior ou igual a)

**Operadores Aritméticos**

Operadores aritméticos são empregados para realizar operações matemáticas em R. Eles são essenciais para realizar cálculos numéricos e manipular dados de forma eficiente dentro de expressões aritméticas.

Abaixo, podemos conhecer alguns exemplos de operadores aritméticos:

* + (adição)
* - (subtração)
* \* (multiplicação)
* / (divisão)
* ^ (exponenciação)
* %% (resto da divisão)
* %/% (divisão inteira)

**Exemplos Práticos**

Agora que fomos apresentados a função ifelse() e a alguns operadores importantes do R, vamos conferir alguns exemplos práticos com eles:

1 - **Operador de Comparação:**

Suponha que você deseje classificar alunos com base em suas notas em uma escala de desempenho, onde notas acima de 70 são consideradas "Aprovado" e as demais são "Reprovado". Utilizando o operador de comparação >= e a função ifelse():

# Exemplo com Operador de Comparação

nota\_aluno <- 78

status\_aprovacao <- ifelse(nota\_aluno >= 70, "Aprovado", "Reprovado")

print(status\_aprovacao)

2 - **Operador Aritmético:**

Vamos considerar um cenário onde você precisa calcular um bônus salarial com base no desempenho do funcionário. O bônus será 10% do salário se a pontuação de desempenho for superior a 80. Utilizando o operador aritmético \* e a função ifelse():

# Exemplo com Operador Aritmético

pontuacao\_desempenho <- 85

salario\_base <- 5000

bonus <- ifelse(pontuacao\_desempenho > 80, salario\_base \* 0.1, salario\_base)

print(bonus)

3 - **Operador Lógico:**

Imagine uma situação em que você deseje verificar se um cliente tem idade superior a 18 anos e possui um status de assinatura ativa para permitir o acesso a um conteúdo restrito. Utilizando operadores lógicos & e == em conjunto com a função ifelse():

# Exemplo com Operador Lógico

idade\_cliente <- 22

status\_assinatura <- **TRUE**

acesso\_permitido <- ifelse(idade\_cliente > 18 & status\_assinatura == **TRUE**, "Acesso Permitido", "Acesso Negado")

print(acesso\_permitido)

A combinação da função ifelse() com uma variedade de operadores proporciona uma poderosa capacidade de controle de fluxo em R. Isso permite a tomada de decisões complexas e a manipulação eficiente de dados com base em condições específicas. A compreensão dessas ferramentas é crucial para o desenvolvimento de scripts robustos e flexíveis em R.

[**Discutir no Fórum**](https://cursos.alura.com.br/forum/curso-r-data-science-conhecendo-linguagem/exercicio-para-saber-mais-operadores-do-r/153972/novo)