```
## ----eval=FALSE-------------------------------------------------------
--------------------------------
## if (condição) {
##   # Bloco de código a ser executado se a condição for verdadeira
## } else {
##   # Bloco de código a ser executado se a condição for falsa
## }


## ---------------------------------------------------------------------
--------------------------------
# Definindo a semente para garantir reprodutibilidade
set.seed(42)

# Gerando um número aleatório entre -10 e 10
numero <- sample(-10:10, 1)

if (numero > 0) {
  print("O número é positivo.")
} else {
  print("O número é negativo ou zero.")
}


## ----eval=FALSE-------------------------------------------------------
--------------------------------
## if (condição1) {
##   # Bloco de código a ser executado se a condição1 for verdadeira
## } else if (condição2) {
##   # Bloco de código a ser executado se a condição2 for verdadeira
## } else {
##   # Bloco de código a ser executado se nenhuma das condições
anteriores for verdadeira
## }


## ---------------------------------------------------------------------
--------------------------------
# Determina a classificação da empresa com base na receita anual
receita_anual <- 1500000

if (receita_anual >= 2000000) {
  print("Empresa de Grande Porte")
} else if (receita_anual >= 1000000) {
  print("Empresa de Médio Porte")
} else if (receita_anual >= 500000) {
  print("Empresa de Pequeno Porte")
} else {
  print("Microempresa")
}


## ---------------------------------------------------------------------
--------------------------------
# Exemplo de loop for para iterar sobre índices
for (i in 1:5) {
  print(i)
```

```
}


## ------------------------------------------------------------------------
------------------------------
# Exemplo de loop for para iterar sobre elementos de um vetor
clientes <- c("João", "Maria", "José", "Ana")
for (nome in clientes) {
  print(nome)
}



## ------------------------------------------------------------------------
------------------------------
set.seed(42)
pib_paises <- runif(10, min = 25000000, max = 40000000)
populacao_paises <- runif(10, min = 1000000, max = 15000000)

pib_per_capita <- numeric(length = 10)

# Loop for para calcular o PIB per capita para cada pais
for (i in 1:10) {
  # Calculando o PIB per capita
  pib_per_capita[i] <- pib_paises[i] / populacao_paises[i]
}
print(round(pib_per_capita, 3))


## ------------------------------------------------------------------------
------------------------------
set.seed(42)
preco_acao <- runif(30, min = 9, max = 15)


## ------------------------------------------------------------------------
------------------------------
media_movel <- numeric(length = 26)  # Vetor para armazenar a média móvel

for (i in 5:30) {
  media_movel[i - 4] <- mean(preco_acao[(i - 4):i])
}


## ------------------------------------------------------------------------
------------------------------
print(media_movel)


## ----echo=FALSE----------------------------------------------------------
------------------------------
ggplot2::ggplot(tidyr::tibble(t=5:30, mm=media_movel), ggplot2::aes(t,
mm)) +
  ggplot2::geom_line(linewidth=2) +
  ggplot2::scale_x_continuous(breaks=seq(5,30,5))+
  ggplot2::labs(x="Dia", y="Média móvel", title="Média Móvel do Preço de
Fechamento")+
  ggplot2::ylim(10, 15)+
```

```
  ggplot2::theme_bw()


## ----eval=FALSE-------------------------------------------------------------
------------------------------
## while (condição) {
##   # Código a ser repetido enquanto a condição for verdadeira
## }


## ---------------------------------------------------------------------------
------------------------------
acoes <- c( "Aprender a programar em R",
        "Aprender a programar em Python",
        "Fazer um café",
        "Descansar")

set.seed(42)
acao <- sample(acoes, 1)
print(acao)


## ---------------------------------------------------------------------------
------------------------------
set.seed(420)
while(acao != "Descansar") {
  acao <- sample(acoes, 1)
  print(acao)
}


## ---------------------------------------------------------------------------
------------------------------
set.seed(42)  # Define uma semente para a replicabilidade dos resultados

# População inicial
populacao <- 1000

# Taxa de crescimento anual da população (em decimal)
taxa_crescimento <- 0.02

# População limite desejada
limite_populacional <- 2000

# Inicializando o contador de anos
anos <- 0

# Simulando o crescimento populacional até atingir o limite
while (populacao < limite_populacional) {
  # Calculando o número de novos indivíduos neste ano
  novos_individuos <- populacao * taxa_crescimento

  # Incrementando a população com os novos indivíduos
  populacao <- populacao + novos_individuos

  # Incrementando o contador de anos
  anos <- anos + 1
}
```

```
# Imprimindo o número de anos necessários para atingir o limite
populacional
print(paste("Foram necessários", anos, "anos para atingir uma população
de", populacao))



## ----eval=FALSE----------------------------------------------------------
-------------------------------
## nome_da_funcao <- function(parametros) {
##   # Corpo da função
##   # Código que realiza a tarefa desejada
##   # Pode incluir operações matemáticas, manipulação de dados, etc.
##   return(resultado)  # Retorna o resultado desejado
## }


## ----------------------------------------------------------------------
-------------------------------
# Função para realizar regressão linear simples
regressao_linear <- function(x, y) {
  modelo <- lm(y ~ x)  # Criando o modelo de regressão linear
  return(modelo)  # Retornando o modelo
}

# Dados de exemplo: salário (y) em função dos anos de educação (x)
anos_educacao <- c(10, 12, 14, 16, 18)
salario <- c(2500, 3300, 3550, 3700, 4500)

# Chamando a função de regressão linear
modelo_regressao <- regressao_linear(anos_educacao, salario)


## ----------------------------------------------------------------------
-------------------------------
# Exibindo os resultados da regressão
summary(modelo_regressao)


## ----echo=FALSE----------------------------------------------------------
-------------------------------
coefs = data.frame(intercept=coef(modelo_regressao)[1],
                   slope=coef(modelo_regressao)[2],
                   class = "coef")

ggplot2::ggplot(tidyr::tibble(anos_educacao = anos_educacao,
       salario = salario),
       ggplot2::aes(anos_educacao, salario))+
  ggplot2::geom_point() +
  ggplot2::geom_abline(data=coefs,
              ggplot2::aes(intercept=intercept,slope=slope,color=class),
              show.legend = TRUE)+
  ggplot2::scale_color_manual(breaks=c("coef"),
                              values="blue", labels=c("Modelo linear"))+
  ggplot2::labs(x="Anos de educação", y="Salário", title="Relação entre
escolaridade e salário",color="")+
  ggplot2::theme_bw()+
```

```
  ggplot2::theme(legend.position = "bottom")


## ----eval=FALSE----------------------------------------------------------
--------------------------------
## install.packages("nome_do_pacote")


## ----eval=FALSE----------------------------------------------------------
--------------------------------
## library(nome_do_pacote)


## ------------------------------------------------------------------------
--------------------------------
set.seed(42)
lancamentos <- rbinom(100, 1, 0.5)


# Exercicios
## ----eval=FALSE----------------------------------------------------------
--------------------------------
## install.packages("nycflights13")


## ----eval=FALSE----------------------------------------------------------
--------------------------------
## library(nycflights13)


## ----eval=FALSE----------------------------------------------------------
--------------------------------
## ?flights
## ?airports
```