```
simu_pop <- function(eq, t, ni, nj, ri, rj){ # criar funcao chamada simu_pop
com os argumentos eq, t, ni, nj, ri e rj
  # conferindo parametros
  if(eq != "exponential" & eq != "logistic") # verificando se o argumento eq
possui as entradas corretas
    {
      stop("Erro em argumento eq. So e possivel simular dinamicas
exponenciais (exponential) e logisticas (logistic)") # se eq nao for um dos
modelos propostos, parar e retonar erro
    }

  if(ni < 0 || nj < 0 || ni > 1 || nj > 1) # verificando se os valores de ni
e nj estao entre 0 e 1
    {
      stop("Erro em argumentos ni ou nj. Valores devem constar entre 0 e 1")
# caso esteja errado, parar simulacao e retornar erro
    }
  if(ri < 1.1 || rj < 1.1 || ri > 3.95 || rj > 3.95) # verificando se os
valores de r estao entre 1.1 e 3.95
    {
      stop("Erro em argumentos ri ou rj. Valores devem constar entre 1.1 e
3.95") # caso esteja errado, parar simulacao e retornar erro
    }

  # parametros conferidos, iniciar simulacao

  data <- as.data.frame(matrix(0, ncol = 3, nrow = length(seq(1, t, 1)))) #
criando matrix vazia de dados
  colnames(data)[1] <- ("tempo") # o nome da coluna 1 e data é tempo
  colnames(data)[2] <- ("ni") # o nome da coluna 2 de data é ni
  colnames(data)[3] <- ("nj") # o nome da coluna 3 de data é nj
  data[,1] <- seq(1, t, 1) # primeira coluna da matriz e o tempo de
simulacao
  data[1,2] <- ni # primeira linha da coluna ni e a populacao inicial
  data[1,3] <- nj # primeira linha da coluna nj e a populacao inicial
  if(eq == "exponential") # condicao para escolher o tipo de modelo, caso o
usuario escolha exponential
    {
    for(i in 2:t) # loop para calcular densidades populacionais no modelo
exponencial
      {
        data[i,2] <- data[i-1,2] + (ri * data[i-1,2]) # a partir da segunda
linha de data, calcular modelo exponencial de pop i ate t
        data[i,3] <- data[i-1,3] + (rj * data[i-1,2]) # a partir da segunda
linha de data, calcular modelo exponencial de pop j ate t
    }
    # gerar grafico com dados simulados do modelo exponencial
    x11() # abrir janela grafica
    par(bg = "gray93", bty = "l", family = "serif") # definindo parametros
graficos
```

```
    pop_graph <- plot(x = data$tempo, y = data$ny, ylim = c(0,1.5), type =
"n",
                        main = "Gráfico da Densidade populacional de i e j",
xlab = "Tempo",
                        ylab = "Dens. populacional") # plotar grafico de
densidade populacional de ni e nj pelo tempo
    lines(x = data$tempo, y = data$ni, col = "red", lwd = 5) # adicionando
linha com dados da populacao i
    lines(x = data$tempo, y = data$nj, col = "blue", lwd = 5) # adicionando
linha com dados da populacao j
    legend(1, 1.5, legend = c("População i", "População j"), col = c("red",
"blue"), lty=1:1, cex=0.7,
           text.font=2, bg='gray85') # adicionando legenda

    return(pop_graph) # retornar grafico final com tempo no eixo X e
densidades pop. no eixo Y
    }
  else # caso contrario, ou seja, caso o modelo escolhido seja o logistico
    {
    for(i in 2:t) # loop para calcular densidades populacionais no modelo
logistico
      {
        data[i,2] <- data[i-1,2] + ((ri * data[i-1,2]) * (1 - data[i-1,2]))
# a partir da segunda linha de data, calcular modelo logistico de pop i ate
t
        data[i,3] <- data[i-1,3] + ((rj * data[i-1,3]) * (1 - data[i-1,3]))
# a partir da segunda linha de data, calcular modelo logistico de pop j ate
t
      }
      # gerar grafico com dados simulados do modelo logistico
      x11() # abrir janela grafica
      par(bg = "gray93", bty = "l", family = "serif") # definindo parametros
graficos
      pop_graph <- plot(x = data$tempo, y = data$ny, ylim = c(0,1.5), type =
"n",
                          main = "Gráfico da Densidade Populacional de i e j",
xlab = "Tempo",
                          ylab = "Dens. populacional") # plotar grafico de
densidade populacional de ni pelo tempo
      lines(x = data$tempo, y = data$ni, col = "red", lwd = 5) # adicionando
linha com dados da populacao i
      lines(x = data$tempo, y = data$nj, col = "blue", lwd = 5) #
adicionando linha com dados da populacao j
      legend(1, 1.5, legend = c("População i", "População j"), col =
c("red", "blue"), lty=1:1, cex=0.7,
             text.font=2, bg='gray85') # adicionando legenda
      return(pop_graph) # retornar grafico final com tempo no eixo X e
densidades pop. no eixo Y
    }
```

```
}
```