```
Algoritmo "SUDOKU"
// Disciplina : Lógica de Programação
// Professor  : Professor Lincoln Fernando
// Descrição  : Jogo de SUDOKU utilizando lógica do VisuAlg
// Autores    : Gabriel V. da Cruz; Gabriel Luis de Santana e João Victor.
// Data atual : 29/05/2023

Var
 l, k : inteiro
 o : vetor[1..9, 1..9] de logico
 h, z, w, reg1, reg2, reg3, i, ent, nuser1, nuser2, nuser3 : inteiro
 pontuser1, pontuser2, pontuser3 : vetor[1..5] de inteiro
 user1, user2, user3 : vetor[1..5] de caractere
 sudoku, sudoku2 : vetor[1..9,1..9] de caractere
inicio

// Início do Procedimento "MENU"
procedimento menu
 Var
  op, op2, q : inteiro
 Inicio
  // Interface do MENU PRINCIPAL
  escreval("   _____ __  _______  _____  __  _ __  __")
  escreval("  / ___ / / / / __ \/ __ \/ /_/ / / / /")
  escreval("  \__ \/ / / / / / / / / / ,< _/ / / / ")
  escreval(" ___/ / /_/ / /_/ / /_/ / /_/ / /| \/ /_/ /  ")
  escreval("/____/\____/_____/\____/_/ |_|\____/   ")
  escreval()
  escreval()
  escreval("1 - JOGAR")
  escreval("2 - EXCLUSÃO DE JOGOS")
  escreval("3 - RANKING")
  escreval("4 - TUTORIAL")
  escreval("5 - SAIR")
  escreval()
  escreva("INSIRA UMA OPÇÃO DESEJADA ->> ")
  leia(op) // Digita a opção desejada do MENU.

  enquanto q = 0 faca // q = 0 torna o procedimento infinito enquanto op
for diferente das opções
   escolha op  // Usa o "escolha" para determinar a opção desejada
    caso 1
     dificuldades // Caso seja digitado 1, começa o jogo.
     jogada
    caso 2
     excluir // Caso seja digitado 2, exibe os jogadores dsiponíveis para
excluir.
    caso 3
     ranking // Caso seja digitado 3, exibe o ranking da dificulade
selecionada.
    caso 4
     tutorial // Caso seja digitado 4, exibe o tutorial para jogar SUDOKU.
    caso 5
```

```
     // Se a opção digitada for 5...
     enquanto q = 0 faca // Pede uma confirmação para sair do programa ou
continuar nele.
      escreval("DESEJA MESMO SAIR DO JOGO?")
      escreval("1 - SIM | 2 - NÃO")
      escreval()
      escreva("->> ")
      leia(op2)
      escolha op2
       caso 1
        limpatela
        escreval("   _____  _______________  ___  ___    ___  ____
")
        escreval("  / __/ |/ / ___/ __/ _ \/ _ \/ _ |  / _ \/ __ \
")
        escreval(" / _/|    / /__/ _|/ , _/ , _/ __ | / / |/ /_/ / _   _
_ ")
        escreval("/___/_/|_/\___/___/_/|_/_/|_/_/ |_|/____/\____/ (_) (_)
(_)")
        fimalgoritmo
       caso 2
        // Caso seja digitado 2 na confirmação, o programa retorna para o
MENU PRINCIPAL.
        interrompa
        limpatela
        menu
      fimescolha
     fimenquanto
    outrocaso
     escreval("INSIRA UMA OPÇÃO VÁLIDA -> ")
     leia(op)
    fimescolha
  fimenquanto
fimprocedimento

// Início do Procedimento "RANKING"
procedimento ranking
 Var
  dificuldade : inteiro
  h : caractere
  valor, val1 : inteiro
  a,b,val : inteiro
  valor2 : caractere
  ruser1, ruser2, ruser3 : vetor[1..5] de caracter
  rpontuser1, rpontuser2, rpontuser3 : vetor[1..5] de inteiro
 Inicio
  // O "para" tem a função de atribuir os cadastros realizados em variáveis
do procedimento "RANKING".
  para a de 1 ate 5 faca
   ruser1[a] <- user1[a]
   ruser2[a] <- user2[a]
   ruser3[a] <- user3[a]
   rpontuser1[a] <- pontuser1[a]
```

```
   rpontuser2[a] <- pontuser2[a]
   rpontuser3[a] <- pontuser3[a]
  fimpara

  limpatela
  // Opções para selecionar a dficuldade do ranking que deseja visualizar.
  escreval("       ____  ___    _   ____ __ _____   _____________ ")
  escreval("     / __ \/   |  / | / /  /_/   _/ | / / ____/ ___/ ")
  escreval("    / /_/ / /| | /  |/ / ,<    / /  |/  / / __ \__ \  ")
  escreval("   / _, _/ ___ |/ /|  / /| |_/ /  /|  / /_/ /___/ /  ")
  escreval("  /_/ |_/_/  |_/_/ |_/_/ |_/___/_/ |_/\____/_____/   ")
  escreval()
  escreval("1 - RANKING DIFICULDADE FÁCIL")
  escreval("2 - RANKING DIFICULDADE MÉDIA")
  escreval("3 - RANKING DIFICULDADE DIFÍCIL")
  escreval()
  escreva("INSIRA A DIFICULDADE CORRESPONDENTE ->> ")
  leia(dificuldade)
  // Enquanto a opção for diferente das disponíveis, aparece mensagem de
opção inválida.
  enquanto (dificuldade <> 1) e (dificuldade <> 2) e (dificuldade <> 3)
faca
   escreva("INSIRA UMA OPÇÃO VÁLIDA ->> ")
   leia(dificuldade)
  fimenquanto
  escolha dificuldade

   caso 1
    // RANKING DA DIFICULDADE FÁCIL
    escreval("Ranking fácil.")
    escreval()

    para a de 1 ate 5 faca //Início da Ordenação da Lista
     para b de 1 ate 5 faca
      val <- b+1
      se val <= 5 entao
       val1 <- rpontuser1[b+1] //Armazena o valor do índice seguinte em
val1
       se (rpontuser1[b] < val1) entao //Verifica se o valor do índice
atual é maior que o próximo
        valor <- rpontuser1[b]  //Inverte os valores entre o índice atual e
o seguinte, caso o atual seja maior
        rpontuser1[b] <- rpontuser1[b+1]
        rpontuser1[b+1] <- valor
        // O nome registrado no cadastro acompanha a ordenação.
        valor2 <- ruser1[b] // Armazena o cadastro em uma variável vazia
para poder mudar a ordem.
        ruser1[b] <- ruser1[b+1]
        ruser1[b+1] <- valor2
       fimse
      fimse
     fimpara
    fimpara
```

```
    // Exibe o Ranking na tela em ordem decrescente, da maior pontuação
para a menor.
    limpatela
    escreval("    ____                __   _
__         ______  __        _ __ ")
    escreval("   / __ \____ _____  / /__(_)___  ____ _   ____ ___  ____
____/ /___     / ____/_/_/_____(_) / ")
    escreval("  / /_/ / __ `/ __ \/  /_/ / __ \/ __ `/  / __ `__ \/ __ \/
__  / __ \   / /_  / __ `/ ___/ / /  ")
    escreval(" / _, _/ /_/ / / / / ,< / / / / / /_/ /  / / / / / / /_/ /
/_/ / /_/ /  / __/ / /_/ / /__/ / /   ")
    escreval("/_/ |_|\__,_/_/ /_/_/|_/_/_/ /_/\__, /  /_/ /_/
/_/\____/\__,_/\____/  /_/    \__,_/\___/_/_/    ")
    escreval("                                /____/
")
    escreval("
")
    escreval("")
    escreval("
|==================/==================|")
    para a de 1 ate 5 faca
     escreval("                        |User: ",ruser1[a]:12,"
Pontuação: ",rpontuser1[a]:4,"|")
     escreval("
|==================/==================|")
    fimpara
    escreval("")
    escreval("")
    // Digita ENTER para retornar ao MENU PRINCIPAL.
    escreva("ENTER PARA VOLTAR AO MENU PRINCIPAL ->> ")
    leia(h)
    limpatela
    menu

   caso 2
    // RANKING DA DIFICULDADE MÉDIA.
    escreval("Ranking Médio.")
    escreval()
    para a de 1 ate 5 faca //Início da Ordenação da Lista
     para b de 1 ate 5 faca
      val <- b+1
      se val <= 5 entao
       val1 <- rpontuser2[b+1] //Armazena o valor do índice seguinte em
val1
       se (rpontuser2[b] < val1) entao //Verifica se o valor do índice
atual é maior que o próximo
        valor <- rpontuser2[b] //Inverte os valores entre o índice atual e
o seguinte, caso o atual seja maior
        rpontuser2[b] <- rpontuser2[b+1]
        rpontuser2[b+1] <- valor
        // O nome registrado acompanha a ordenação.
```

```
        valor2 <- ruser2[b] // Armazena o cadastro em uma variável vazia
para poder mudar a ordem.
        ruser2[b] <- ruser2[b+1]
        ruser2[b+1] <- valor2
       fimse
      fimse
     fimpara
    fimpara
    limpatela

    // Exibe o Ranking na tela em ordem decrescente, da maior pontuação
para a menor.
    escreval("    ____                __   _
__          __  ___ __       ___       ")
    escreval("   / __ \____ _____  / /__(_)___  ____ _   ____ ___  ____
____/ /___     /  |/  / _ ____/ (_)___  ")
    escreval("  / /_/ / __ `/ __ \/  /_/ / __ \/ __ `/  / __ `__ \/ __ \/
__  / __ \   / /|_/ / _ \/ __  / / __ \ ")
    escreval(" / _, _/ /_/ / / / / ,< / / / / / /_/ /  / / / / / / /_/ /
/_/ / /_/ /  / /  / /  __/ /_/ / / /_/ / ")
    escreval("/_/ |_|\__,_/_/ /_/_/|_/_/_/ /_/\__, /  /_/ /_/
/_/\____/\__,_/\____/  /_/  /_/\___/\__,_/_/\____/  ")
    escreval("                                  /____/
")
    escreval("
")
    escreval("")
    escreval("
|==================/==================|")
    para a de 1 ate 5 faca
     escreval("                        |User: ",ruser2[a]:12,"
Pontuação: ",rpontuser2[a]:4,"|")
     escreval("
|==================/==================|")
    fimpara
    escreval("")
    escreval("")
    // Digita ENTER para retornar ao MENU PRINCIPAL.
    escreva("ENTER PARA VOLTAR AO MENU PRINCIPAL ->> ")
    leia(h)
    limpatela
    menu

   caso 3
    // RANKING DA DIFICULDADE DIFÍCIL.
    escreval("Ranking Difícil.")
    escreval()
    para a de 1 ate 5 faca //Início da Ordenação da Lista
     para b de 1 ate 5 faca
      val <- b+1
      se val <= 5 entao
       val1 <- rpontuser3[b+1] //Armazena o valor do índice seguinte em
val1
```

```
       se (rpontuser3[b] < val1) entao //Verifica se o valor do índice
atual é maior que o próximo
        valor <- rpontuser3[b] //Inverte os valores entre o índice atual e
o seguinte, caso o atual seja maior
        rpontuser3[b] <- rpontuser3[b+1]
        rpontuser3[b+1] <- valor
        // O nome registrado acompanha a ordenação.
        valor2 <- ruser3[b] // Armazena o cadastro em uma variável vazia
para poder mudar a ordem.
        ruser3[b] <- ruser3[b+1]
        ruser3[b+1] <- valor2
       fimse
      fimse
     fimpara
    fimpara
    limpatela

    // Exibe o Ranking na tela em ordem decrescente, da maior pontuação
para a menor.
    escreval("    ____                __   _
__         ____  _ ______     _ __  ")
    escreval("   / __ \____ _____  / /__(_)___  ____ _   ____ ___  ____
____/ /___     / __ \(_) __/_/____(_) /  ")
    escreval("  / /_/ / __ `/ __ \/  /_/ / __ \/ __ `/  / __ `__ \/ __ \/
__  / __ \   / / / / / /_/ / ___/ / /   ")
    escreval(" / _, _/ /_/ / / / / ,< / / / / / /_/ /  / / / / / / /_/ /
/_/ / /_/ /  / /_/ / / __/ / /__/ / /    ")
    escreval("/_/ |_|\__,_/_/ /_/_/|_/_/_/ /_/\__, /  /_/ /_/
/_/\____/\__,_/\____/  /_____/_/_/ /_/\___/_/_/     ")
    escreval("                                  /____/
")
    escreval("
")
    escreval("")
    escreval("
|==================/==================|")
    para a de 1 ate 5 faca
     escreval("                          |User: ",ruser3[a]:12,"
Pontuação: ",rpontuser3[a]:4,"|")
     escreval("
|==================/==================|")
    fimpara
    escreval("")
    escreval("")
    // Digita ENTER para retornar ao MENU PRINCIPAL.
    escreva("ENTER PARA VOLTAR AO MENU PRINCIPAL ->> ")
    leia(h)
    limpatela
    menu
   fimescolha
fimprocedimento
```

```
// Início do Procedimento "TUTORIAL"
procedimento tutorial
 Inicio
  limpatela
  // Exibe o texto do tutorial.
  escreval("
_____________________________________________________________________________
_____________________________")
  escreval("   |
|")
  escreval("   |
|")
  escreval("   |                  ______                                  _
|")
  escreval("   |                / ____/___  ____ ___  ____       (_)___
____ _____ ______   _                                  |")
  escreval("   |               / /   / __ \/ __ `__ \/ __ \     / / __ \/
__ `/ __ `/ ___/  (_)                          |")
  escreval("   |              / /___/ /_/ / / / / / / / /_/ /    / / /_/ /
/_/ / /_/ / /     _                             |")
  escreval("   |              \____/\____/_/ /_/ /_/\____/  __/ /\____/\__,
/\__,_/_/     (_)                           |")
  escreval("   |                                                /___/
/____/                                      |")
  escreval("   |
|")
  escreval("   |         O objetivo é completar a tabela com os números de
1 a 9, sem repetir nenhum por co-           |")
  escreval("   |         luna, linha ou bloco. A cada novo jogo, a tabela
estará parcialmente preenchida  e           |")
  escreval("   |         terá de usar esses dígitos para encontrar a
solução das demais quadrículas em branco.         |")
  escreval("   |
|")
  escreval("   |           Para começar a jogar terá apenas de escolher um
dos três níveis de dificuldade disponíveis: |")
  escreval("   |            - Fácil
|")
  escreval("   |            - Médio
|")
  escreval("   |            - Difícil
|")
  escreval("   |
|")
  escreval("   |          Quanto mais difícil o nível, menos preenchida
estará a tabela ao início do jogo.            |")
  escreval("   |
|")
```

```
  escreval("
|_____________________________________________________________________________
_____________________________ |")
  escreval("   |
|")
  escreval("   |
|")
  escreval("   |                              ____
|")
  escreval("   |                           / __ \___  ____ _____________ ______
_                                            |")
  escreval("   |                          / /_/ / _ \/ __ `/ ___/ __ `/ ___/
(_)                                        |")
  escreval("   |                         / _, _/  __/ /_/ / /  / /_/ (__  )  _
|")
  escreval("   |                        /_/ |_|\___/\__, /_/   \__,_/____/
(_)                                              |")
  escreval("   |                                  /____/
|")
  escreval("   |
|")
  escreval("   |    - A tabela do Sudoku é composta por 9x9 espaços.
|")
  escreval("   |    - Só pode usar números de 1 a 9.
|")
  escreval("   |    - Cada bloco, linha e coluna pode apenas conter números
de 1 a 9.                                  |")
  escreval("   |    - A cada acerto do número correto no tabuleiro, o
usuário ganha 10 pontos e a cada erro perde 10.  |")
  escreval("   |    - Cada número no bloco 3×3, na coluna vertical ou na
linha horizontal só pode ser usado uma vez.   |")
  escreval("   |    - O jogo termina quando toda a tabela do Sudoku estiver
preenchida corretamente com números.       |")
  escreval("   |
|")
  escreval("
|_____________________________________________________________________________
_____________________________|")
  escreval()
  escreval()
  // Digita ENTER para retornar ao MENU PRINCIPAL.
  escreva("   ENTER PARA VOLTAR AO MENU PRINCIPAL ->> ")
  leia(ent)
  limpatela
  menu
fimprocedimento


// Início do Procedimento "CADASTRO - NÍVEL FÁCIL"
procedimento cadastrof
 Inicio
  limpatela
```

```
  // O "para" tem a função de atribuir "N/A" para jogadores não
cadastrados.
  para i de 1 ate 5 faca
   se (user1[i] = "") ou (user1[i] = " ") entao
    user1[i] <- "N/A"
   fimse
  fimpara

  escreval("      ______             __              __                ")
  escreval("     / ____/___  ____  / /___  _____/ /_____________   ")
  escreval("    / /   / __ `/ __  / __ `/ ___/ __/ ___/ __ \  ")
  escreval("   / /___/ /_/ / /_/ / /_/ (__  ) /_/ /  / /_/ /  ")
  escreval("   \____/\__,_/\__,_/\__,_/____/\__/_/   \____/   ")
  escreval("                        ___  __      _ __          ")
  escreval("                  / _/_/_/____(_) /    ")
  escreval("                 / _/ _ `/ __/ / /     ")
  escreval("                /_/ \_,_/\__/_/_/      ")
  escreval()
  escreval(" SELECIONE O JOGADOR QUE DESEJA CADASTRAR:")
  para nuser1 de 1 ate 5 faca
   escreval(" JOGADOR ", nuser1, ": ", user1[nuser1])
  fimpara
  escreval(" DIGITE 0 PARA VOLTAR AO MENU")
  escreva(" -> ")
  // Pede para digitar um dos 5 espaços disponíveis para cadastro de
jogador na dificuldade fácil.
  leia(nuser1)
  // Caso a opção seja 0 o programa retorna ao MENU PRINCIPAL.
  se nuser1 = 0 entao
   limpatela
   menu
  fimse
  // Enquanto o número digitado for menor que 0 ou maior que 5 ele pede
para reinserir.
  enquanto (nuser1 > 5) ou (nuser1 < 0) faca
   escreva("INSIRA UM NÚMERO DE JOGADOR VÁLIDO: ")
   leia(nuser1)
   se nuser1 = 0 entao
    limpatela
    menu
   fimse
  fimenquanto
  // Caso o número seja correspondente a um cadastro existente,
  // o algoritmo pergunta se deseja continuar com aquele cadastro
  // em um novo jogo.
  enquanto (user1[nuser1] <> "") e (user1[nuser1] <> "N/A") faca
   escreval("ESTE JOGADOR JÁ FOI CADASTRADO!")
   escreval("DESEJA JOGAR COM ESSE CADASTRO? ")
   escreval("1 <<- SIM")
   escreval("2 <<- NÃO")
   leia(h)
   enquanto (h <> 1) e (h <> 2) FACA
    escreval("INSIRA UMA OPÇÃO VÁLIDA")
```

```
    leia(h)
   fimenquanto
   se h = 1 entao
    limpatela
    jogada
   senao
    // Senão o algoritmo repete a função "CADASTROF".
    se h = 2 entao
     limpatela
     cadastrof
    fimse
   fimse
  fimenquanto
  // Insira o NOME DO JOGADOR.
  escreva("INSIRA O NOME DO JOGADOR: ")
  leia(user1[nuser1])
  // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
  enquanto (user1[nuser1] = "") ou (user1[nuser1] = " ") faca
   escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
   escreva("INSIRA UM NOME DIFERENTE: ")
   leia(user1[nuser1])
  fimenquanto
  // Torna os caracteres maiúsculos.
  user1[nuser1] <- maiusc(user1[nuser1])

  escolha nuser1
   caso 1
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user1[1] = user1[2]) ou (user1[1] = user1[3]) ou (user1[1] =
user1[4]) ou (user1[1] = user1[5]) entao
     enquanto (user1[1] = user1[2]) ou (user1[1] = user1[3]) ou (user1[1] =
user1[4]) ou (user1[1] = user1[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user1[1])
      user1[1] <- maiusc(user1[1])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user1[1] = "") ou (user1[1] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user1[1])
       user1[1] <- maiusc(user1[1])
      fimenquanto
     fimenquanto
    fimse
   caso 2
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user1[2] = user1[1]) ou (user1[2] = user1[3]) ou (user1[2] =
user1[4]) ou (user1[2] = user1[5]) entao
```

```
     enquanto (user1[2] = user1[1]) ou (user1[2] = user1[3]) ou (user1[2] =
user1[4]) ou (user1[2] = user1[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user1[2])
      user1[2] <- maiusc(user1[2])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user1[2] = "") ou (user1[2] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user1[2])
       user1[2] <- maiusc(user1[2])
      fimenquanto
     fimenquanto
    fimse
   caso 3
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user1[3] = user1[1]) ou (user1[3] = user1[2]) ou (user1[3] =
user1[4]) ou (user1[3] = user1[5]) entao
     enquanto (user1[3] = user1[1]) ou (user1[3] = user1[2]) ou (user1[3] =
user1[4]) ou (user1[3] = user1[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user1[3])
      user1[3] <- maiusc(user1[3])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user1[3] = "") ou (user1[3] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user1[3])
       user1[3] <- maiusc(user1[3])
      fimenquanto
     fimenquanto
    fimse
   caso 4
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user1[4] = user1[1]) ou (user1[4] = user1[2]) ou (user1[4] =
user1[3]) ou (user1[4] = user1[5]) entao
     enquanto (user1[4] = user1[1]) ou (user1[4] = user1[2]) ou (user1[4] =
user1[3]) ou (user1[4] = user1[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user1[4])
      user1[4] <- maiusc(user1[4])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user1[4] = "") ou (user1[4] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
```

```
       leia(user1[4])
       user1[4] <- maiusc(user1[4])
      fimenquanto
     fimenquanto
    fimse
   caso 5
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user1[5] = user1[1]) ou (user1[5] = user1[2]) ou (user1[5] =
user1[3]) ou (user1[5] = user1[4]) entao
     enquanto (user1[5] = user1[1]) ou (user1[5] = user1[2]) ou (user1[5] =
user1[3]) ou (user1[5] = user1[4]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user1[5])
      user1[5] <- maiusc(user1[5])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user1[5] = "") ou (user1[5] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user1[5])
       user1[5] <- maiusc(user1[5])
      fimenquanto
     fimenquanto
    fimse
  fimescolha
fimprocedimento

// Início do Procedimento "CADASTRO - NÍVEL MÉDIO"
procedimento cadastrom
 Inicio
  limpatela
  // O "para" tem a função de atribuir "N/A" para jogadores não
cadastrados.
  para i de 1 ate 5 faca
   se (user2[i] = "") ou (user2[i] = " ") entao
    user2[i] <- "N/A"
   fimse
  fimpara
  escreval("      ______          __          __                ")
  escreval("     / ____/___  ____/ /___  _____/ /_____________   ")
  escreval("    / /   / __ `/ __  / __ `/ ___/ __/ ___/ __ \  ")
  escreval("   / /___/ /_/ / /_/ / /_/ (__  ) /_/ /  / /_/ /  ")
  escreval("   \____/\__,_/\__,_/\__,_/____/\__/_/   \____/   ")
  escreval("                    __  ___ __     ___")
  escreval("                  /  |/ /./_/___/ (_)__   ")
  escreval("                 / /|_/ / -_) _  / / _ \  ")
  escreval("                /_/  /_/\__/\_,_/_/\___/  ")
  escreval()
  escreval(" SELECIONE O JOGADOR QUE DESEJA CADASTRAR:")
  para nuser2 de 1 ate 5 faca
   escreval(" Jogador ", nuser2, ": ", user2[nuser2])
```

```
  fimpara
  escreval(" DIGITE 0 PARA VOLTAR AO MENU")
  escreva(" -> ")
  // Pede para digitar um dos 5 espaços disponíveis para cadastro de
jogador na dificuldade média.
  leia(nuser2)
  // Caso a opção seja 0 o programa retorna ao MENU PRINCIPAL.
  se nuser2 = 0 entao
   limpatela
   menu
  fimse
  // Enquanto o número digitado for menor que 0 ou maior que 5 ele pede
para reinserir.
  enquanto (nuser2 > 5) ou (nuser2 < 0) faca
   escreva("INSIRA UM NÚMERO DE JOGADOR VÁLIDO: ")
   leia(nuser2)
   se nuser2 = 0 entao
    limpatela
    menu
   fimse
  fimenquanto
  // Caso o número seja correspondente a um cadastro existente,
  // o algoritmo pergunta se deseja continuar com aquele cadastro
  // em um novo jogo.
  enquanto (user2[nuser2] <> "") e (user2[nuser2] <> "N/A") faca
   escreval("ESTE JOGADOR JÁ FOI CADASTRADO!")
   escreval("DESEJA JOGAR COM ESSE CADASTRO? ")
   escreval("1 <<- SIM")
   escreval("2 <<- NÃO")
   leia(h)
   enquanto (h <> 1) e (h <> 2) FACA
    escreval("INSIRA UMA OPÇÃO VÁLIDA")
    leia(h)
   fimenquanto
   se h = 1 entao
    limpatela
    jogada
   senao
    // Senão o algoritmo repete a função "CADASTROM".
    se h = 2 entao
     limpatela
     cadastrom
    fimse
   fimse
  fimenquanto
  // Insira o NOME DO JOGADOR.
  escreva("INSIRA O NOME DO JOGADOR: ")
  leia(user2[nuser2])
  // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
  enquanto (user2[nuser2] = "") ou (user2[nuser2] = " ") faca
   escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
   escreva("INSIRA UM NOME DIFERENTE: ")
```

```
   leia(user2[nuser2])
  fimenquanto
  // Torna os caracteres maiúsculos.
  user2[nuser2] <- maiusc(user2[nuser2])

  escolha nuser2
   caso 1
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user2[1] = user2[2]) ou (user2[1] = user2[3]) ou (user2[1] =
user2[4]) ou (user2[1] = user2[5]) entao
     enquanto (user2[1] = user2[2]) ou (user2[1] = user2[3]) ou (user2[1] =
user2[4]) ou (user2[1] = user2[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user2[1])
      user2[1] <- maiusc(user2[1])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user2[1] = "") ou (user2[1] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user2[1])
       user2[1] <- maiusc(user2[1])
      fimenquanto
     fimenquanto
    fimse
   caso 2
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user2[2] = user2[1]) ou (user2[2] = user2[3]) ou (user2[2] =
user2[4]) ou (user2[2] = user2[5]) entao
     enquanto (user2[2] = user2[1]) ou (user2[2] = user2[3]) ou (user2[2] =
user2[4]) ou (user2[2] = user2[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user2[2])
      user2[2] <- maiusc(user2[2])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user2[2] = "") ou (user2[2] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user2[2])
       user2[2] <- maiusc(user2[2])
      fimenquanto
     fimenquanto
    fimse
   caso 3
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user2[3] = user2[1]) ou (user2[3] = user2[2]) ou (user2[3] =
user2[4]) ou (user2[3] = user2[5]) entao
```

```
     enquanto (user2[3] = user2[1]) ou (user2[3] = user2[2]) ou (user2[3] =
user2[4]) ou (user2[3] = user2[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user2[3])
      user2[3] <- maiusc(user2[3])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user2[3] = "") ou (user2[3] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user2[3])
       user2[3] <- maiusc(user2[3])
      fimenquanto
     fimenquanto
    fimse
   caso 4
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user2[4] = user2[1]) ou (user2[4] = user2[2]) ou (user2[4] =
user2[3]) ou (user2[4] = user2[5]) entao
     enquanto (user2[4] = user2[1]) ou (user2[4] = user2[2]) ou (user2[4] =
user2[3]) ou (user2[4] = user2[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user2[4])
      user2[4] <- maiusc(user2[4])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user2[4] = "") ou (user2[4] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user2[4])
       user2[4] <- maiusc(user2[4])
      fimenquanto
     fimenquanto
    fimse
   caso 5
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user2[5] = user2[1]) ou (user2[5] = user2[2]) ou (user2[5] =
user2[3]) ou (user2[5] = user2[4]) entao
     enquanto (user2[5] = user2[1]) ou (user2[5] = user2[2]) ou (user2[5] =
user2[3]) ou (user2[5] = user2[4]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user2[5])
      user2[5] <- maiusc(user2[5])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user2[5] = "") ou (user2[5] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
```

```
       leia(user2[5])
       user2[5] <- maiusc(user2[5])
      fimenquanto
     fimenquanto
    fimse
  fimescolha
fimprocedimento

// Início do Procedimento "CADASTRO - NÍVEL DIFÍCIL"
procedimento cadastrod
 Inicio
  limpatela
  // O "para" tem a função de atribuir "N/A" para jogadores não
cadastrados.
  para i de 1 ate 5 faca
   se (user3[i] = "") ou (user3[i] = " ") entao
    user3[i] <- "N/A"
   fimse
  fimpara
  escreval("      ______          __          __                ")
  escreval("     / ____/___  ____/ /___  _____/ /__________    ")
  escreval("    / /   / __ `/ __  / __ `/ ___/ __/ ___/ __ \  ")
  escreval("   / /___/ /_/ / /_/ / /_/ (__  ) /_/ /  / /_/ /  ")
  escreval("   \____/\__,_/\__,_/\__,_/____/\__/_/   \____/   ")
  escreval("                 ___  _ ____     _ __ ")
  escreval("              / _ \(_) _(_)_  (_) / ")
  escreval("             / /  / / _/ / _\/ / / ")
  escreval("            /____/_/_/ _/\__/_/_/  ")
  escreval()
  escreval(" SELECIONE O JOGADOR QUE DESEJA CADASTRAR:")
  para nuser3 de 1 ate 5 faca
   escreval(" JOGADOR ", nuser3, ": ", user3[nuser3])
  fimpara
  escreval(" DIGITE 0 PARA VOLTAR AO MENU")
  escreva(" -> ")
  // Pede para digitar um dos 5 espaços disponíveis para cadastro de
jogador na dificuldade difícil.
  leia(nuser3)
  // Caso a opção seja 0 o programa retorna ao MENU PRINCIPAL.
  se nuser3 = 0 entao
   limpatela
   menu
  fimse
  // Enquanto o número digitado for menor que 0 ou maior que 5 ele pede
para reinserir.
  enquanto (nuser3 > 5) ou (nuser3 < 0) faca
   escreva("INSIRA UM NÚMERO DE JOGADOR VÁLIDO: ")
   leia(nuser3)
   se nuser3 = 0 entao
    limpatela
    menu
   fimse
  fimenquanto
```

```
  // Caso o número seja correspondente a um cadastro existente,
  // o algoritmo pergunta se deseja continuar com aquele cadastro
  // em um novo jogo.
  enquanto (user3[nuser3] <> "") e (user3[nuser3] <> "N/A") faca
   escreval("ESTE JOGADOR JÁ FOI CADASTRADO!")
   escreval("DESEJA JOGAR COM ESSE CADASTRO? ")
   escreval("1 <<- SIM")
   escreval("2 <<- NÃO")
   leia(h)
   enquanto (h <> 1) e (h <> 2) FACA
    escreval("INSIRA UMA OPÇÃO VÁLIDA")
    leia(h)
   fimenquanto
   se h = 1 entao
    limpatela
    jogada
   senao
    // Senão o algoritmo repete a função "CADASTROD".
    se h = 2 entao
     limpatela
     cadastrod
    fimse
   fimse
  fimenquanto
  // Insira o NOME DO JOGADOR.
  escreva("INSIRA O NOME DO JOGADOR: ")
  leia(user3[nuser3])
  // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
  enquanto (user3[nuser3] = "") ou (user3[nuser3] = " ") faca
   escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
   escreva("INSIRA UM NOME DIFERENTE: ")
   leia(user3[nuser3])
  fimenquanto
  // Torna os caracteres maiúsculos.
  user3[nuser3] <- maiusc(user3[nuser3])

  escolha nuser3
   caso 1
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user3[1] = user3[2]) ou (user3[1] = user3[3]) ou (user3[1] =
user2[4]) ou (user3[1] = user3[5]) entao
     enquanto (user3[1] = user3[2]) ou (user3[1] = user3[3]) ou (user3[1] =
user3[4]) ou (user3[1] = user3[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user3[1])
      user3[1] <- maiusc(user3[1])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user3[nuser3] = "") ou (user3[nuser3] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
```

```
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user3[nuser3])
       user3[1] <- maiusc(user3[1])
      fimenquanto
     fimenquanto
    fimse
   caso 2
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user3[2] = user3[1]) ou (user3[2] = user3[3]) ou (user3[2] =
user3[4]) ou (user3[2] = user3[5]) entao
     enquanto (user3[2] = user3[1]) ou (user3[2] = user3[3]) ou (user3[2] =
user3[4]) ou (user3[2] = user3[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user3[2])
      user3[2] <- maiusc(user3[2])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user3[nuser3] = "") ou (user3[nuser3] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user3[nuser3])
       user3[2] <- maiusc(user3[2])
      fimenquanto
     fimenquanto
    fimse
   caso 3
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user3[3] = user3[1]) ou (user3[3] = user3[2]) ou (user3[3] =
user3[4]) ou (user3[3] = user3[5]) entao
     enquanto (user3[3] = user3[1]) ou (user3[3] = user3[2]) ou (user3[3] =
user3[4]) ou (user3[3] = user3[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user3[3])
      user3[3] <- maiusc(user3[3])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user3[nuser3] = "") ou (user3[nuser3] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user3[nuser3])
       user3[3] <- maiusc(user3[3])
      fimenquanto
     fimenquanto
    fimse
   caso 4
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user3[4] = user3[1]) ou (user3[4] = user3[2]) ou (user3[4] =
user3[3]) ou (user3[4] = user3[5]) entao
```

```
     enquanto (user3[4] = user3[1]) ou (user3[4] = user3[2]) ou (user3[4] =
user3[3]) ou (user3[4] = user3[5]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user3[4])
      user3[4] <- maiusc(user3[4])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user3[nuser3] = "") ou (user3[nuser3] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user3[nuser3])
       user3[4] <- maiusc(user3[4])
      fimenquanto
     fimenquanto
    fimse
   caso 5
    // Se o nome inserido for igual outro existente, pede para inserir
outro nome.
    se (user3[5] = user3[1]) ou (user3[5] = user3[2]) ou (user3[5] =
user3[3]) ou (user3[5] = user3[4]) entao
     enquanto (user3[5] = user3[1]) ou (user3[5] = user3[2]) ou (user3[5] =
user3[3]) ou (user3[5] = user3[4]) faca
      escreval(" ESTE NOME DE JOGADOR JÁ FOI CADASTRADO!")
      escreva("INSIRA UM NOME DIFERENTE: ")
      leia(user3[5])
      user3[5] <- maiusc(user3[5])
      // Caso o nome esteja em branco, aparece a mensagem para inserir
novamente.
      enquanto (user3[nuser3] = "") ou (user3[nuser3] = " ") faca
       escreval("SEU NOME NÃO PODE ESTAR EM BRANCO!")
       escreva("INSIRA UM NOME DIFERENTE: ")
       leia(user3[nuser3])
       user3[5] <- maiusc(user3[5])
      fimenquanto
     fimenquanto
    fimse
  fimescolha
fimprocedimento



// Início do Procedimento "EXCLUIR"
procedimento excluir
 Var
  k, q, op, op2, op3 : inteiro
 Inicio
  limpatela
  // Exibe os jogadores de cada dificuldade.
  escreval("    ______          __      _             __
")
  escreval("   / ____/  _______/ /_  __(_)____       / /___  ____ _____
_____ ")
```

```
  escreval("  / __/ | |/_/ ___/ / / / / / ___/  __  / / __ \/ __ `/ __ \/
___/ ")
  escreval(" / /____>  </ /__/ / /_/ / / /     / /_/ / /_/ / /_/ / /_/ (__
)  ")
  escreval("/_____/_/|_|\___/_/\__,_/_/_/      \____/\____/\__,
/\____/____/   ")
  escreval("                                              /____/
")
  escreval()

  escreval("+==================+==================+==================+")
  escreval("|      FÁCIL       |       MÉDIO      |      DIFÍCIL     |")
  escreval("|__________________|__________________|__________________|")
  escreval("|                  |                  |                  |")
  // O "para" tem a função de atribuir "N/A" para jogadores não
cadastrados.
  para i de 1 ate 5 faca
   se user1[i] = "" entao
    user1[i] <- "N/A"
   fimse
   se user2[i] = "" entao
    user2[i] <- "N/A"
   fimse
   se user3[i] = "" entao
    user3[i] <- "N/A"
   fimse
   escreval("| ",user1[i]:12," ",pontuser1[i]:4," | ", user2[i]:12,"
",pontuser2[i]:4," | ", user3[i]:12," ", pontuser3[i]:4," |")
  fimpara
  escreval("|__________________|__________________|__________________|")

  escreval()
  escreval("0 -> MENU")
  escreval("1 -> FÁCIL")
  escreval("2 -> MÉDIO")
  escreval("3 -> DIFÍCIL")
  escreval()
  // Pede para inserir a dificuldade onde o jogador desejado se encontra.
  escreva("INSIRA A DIFICULDADE DO JOGO ->> ")
  leia(op)
  enquanto op <> 0 faca
   escolha op

    caso 1
     // Dificuldade Fácil
     // Insere o número para o jogador desejado.
     escreva("INSIRA O NÚMERO DO JOGADOR CORRESPONDENTE ->> ")
     leia(op2)
     // Enquanto o número for menor que 1 ou maior que 5, pede para
reinserir.
     enquanto (op2 < 1) ou (op2 > 5) faca
      escreva("INSIRA UM NÚMERO DE JOGADOR EXISTENTE ->> ")
      leia(op2)
```

```
     fimenquanto
     // O cadastro do jogador é apagado e sua pontuação retorna a 0.
     user1[op2] <- ""
     pontuser1[op2] <- 0
     escreval()
     escreval("JOGADOR N°",op2," EXCLUÍDO")
     escreval()
     escreval("=============================================")
     escreval()
     escreval("1 -> SIM")
     escreval("2 -> NÃO")
     escreval()
     // Pergunta se deseja excluir outro jogador, ou retornar ao MENU
PRINCIPAL.
     escreva("DESEJA EXCLUIR OUTRO JOGADOR? ->> ")
     leia(op3)
     // Enquanto op3 for menor que 1 ou maior que 2, pede para inserir
opção válida.
     enquanto (op3 < 1) ou (op3 > 2) faca
      escreva("DESEJA EXCLUIR OUTRO JOGADOR? ->> ")
      leia(op3)
     fimenquanto
     escolha op3
      caso 1
       excluir // Repete o procedimento "EXCLUIR".
      caso 2
       interrompa
       limpatela
       menu // Retorna ao MENU PRINCIPAL.
     fimescolha

    caso 2
     // Dificuldade Média.
     // Insere o número para o jogador desejado.
     escreva("INSIRA O NÚMERO DO JOGADOR CORRESPONDENTE ->> ")
     leia(op2)
     // Enquanto o número for menor que 1 ou maior que 5, pede para
reinserir.
     enquanto (op2 < 1) ou (op2 > 5) faca
      escreva("INSIRA UM NÚMERO DE JOGADOR EXISTENTE ->> ")
      leia(op2)
     fimenquanto
     // O cadastro do jogador é apagado e sua pontuação retorna a 0.
     user2[op2] <- ""
     pontuser2[op2] <- 0
     escreval()
     escreval("JOGADOR N°",op2," EXCLUÍDO")
     escreval()
     escreval("=============================================")
     escreval()
     escreval("1 -> SIM")
     escreval("2 -> NÃO")
```

```
     // Pergunta se deseja excluir outro jogador, ou retornar ao MENU
PRINCIPAL.
     escreva("DESEJA EXCLUIR OUTRO JOGADOR? ->> ")
     leia(op3)
     // Enquanto op3 for menor que 1 ou maior que 2, pede para inserir
opção válida.
     enquanto (op3 < 1) ou (op3 > 2) faca
      escreva("DESEJA EXCLUIR OUTRO JOGADOR? ->> ")
      leia(op3)
     fimenquanto
     escolha op3
      caso 1
       excluir // Repete o procedimento "EXCLUIR".
      caso 2
       interrompa
       limpatela
       menu // Retorna ao MENU PRINCIPAL.
     fimescolha

    caso 3
     // Dificuldade Difícil
     // Insere o número para o jogador desejado.
     escreva("INSIRA O NÚMERO DO JOGADOR CORRESPONDENTE ->> ")
     leia(op2)
     // Enquanto o número for menor que 1 ou maior que 5, pede para
reinserir.
     enquanto (op2 < 1) ou (op2 > 5) faca
      escreva("INSIRA UM NÚMERO DE JOGADOR EXISTENTE ->> ")
      leia(op2)
     fimenquanto
     // O cadastro do jogador é apagado e sua pontuação retorna a 0.
     user3[op2] <- ""
     pontuser3[op2] <- 0
     escreval()
     escreval("JOGADOR N°",op2," EXCLUÍDO")
     escreval()
     escreval("=============================================")
     escreval()
     escreval("1 -> SIM")
     escreval("2 -> NÃO")
     // Pergunta se deseja excluir outro jogador, ou retornar ao MENU
PRINCIPAL.
     escreva("DESEJA EXCLUIR OUTRO JOGADOR? ->> ")
     leia(op3)
     // Enquanto op3 for menor que 1 ou maior que 2, pede para inserir
opção válida.
     enquanto (op3 < 1) ou (op3 > 2) faca
      escreva("DESEJA EXCLUIR OUTRO JOGADOR? ->> ")
      leia(op3)
     fimenquanto
     escolha op3
      caso 1
       excluir // Repete o procedimento "EXCLUIR".
```

```
      caso 2
       interrompa
       limpatela
       menu // Retorna ao MENU PRINCIPAL.
      fimescolha

     outrocaso
      // Enquanto a dificuldade selecionada
      // for diferente das opções disponíveis,
      // é solicitado para inserir novamente.
      escreva("INSIRA A DIFICULDADE DO JOGO ->> ")
      leia(op)
    fimescolha
  fimenquanto
  limpatela
  menu
fimprocedimento


// Início do Procedimento "JOGADA"
procedimento jogada
 Var
  a, b, d, s, x, y, k : inteiro
  c : caractere
 Inicio
 // Exibe a dificuldade selecionada no procedimento "DIFICULDADES".
 se w = 1 entao
  escreval("            ____  __      _ __")
  escreval("          / __/_/_/____(_) /")
  escreval("         / _|/ _ `/ __/ / / ")
  escreval("        /_/  \_,_/\__/_/_/  ")
 senao

  se w = 2 entao
   escreval("           __  ___ __     ___     ")
   escreval("          /  |/  |/_/___/ (_)__ ")
   escreval("         / /|_/ / -_) _  / / _ \")
   escreval("        /_/  /_/\__/\_,_/_/\___/")
  senao

   se w = 3 entao
    escreval("           ___  _ _____    _ __")
    escreval("          / _ \(_) _/_/___(_) /")
    escreval("         / /| / / _/ / __/ / / ")
    escreval("        /____/_/_/|_/\__/_/_/  ")
   fimse
  fimse
 fimse

 // Torna o jogo infinito até estar completo, ou digitar 10 para retornar
ao MENU PRINCIPAL.
 enquanto d = 0 faca
  // Variável "l" tem a função de contar a quantidade de casas preenchidas;
```

```
  // Caso seja igual a 81, o jogo encerrará;
  l <- 0
  para i de 1 ate 9 faca
   para z de 1 ate 9 faca
    se sudoku[z,i] = sudoku2[z,i] entao
     // Para cada espaço preenchido corretamente, "l" soma mais um, até
chegar em 81;
     // Espaço do vetor "o" assume verdadeiro caso o espaço esteja
preenchido;
     o[z,i] <- verdadeiro
     l <- l + 1
    senao
     // Ou assume falso caso não esteja.
     o[z,i] <- falso
    fimse
   fimpara
  fimpara

  escreval()
  // Exibe o tabuleiro do SUDOKU a cada nova jogada.
  escreval("     |-----------------------------------|")
  escreval("       C1  C2  C3  C4  C5  C6  C7  C8  C9 ")
  escreval("| -- |-----------------------------------|")
  escreval("| L1 | ",sudoku[1,1]," | ",sudoku[2,1]," | ",sudoku[3,1]," |
",sudoku[4,1]," | ",sudoku[5,1]," | ",sudoku[6,1]," | ",sudoku[7,1]," |
",sudoku[8,1]," | ",sudoku[9,1]," |")
  escreval("|    | - - - - - | - - - - - | - - - - - |")
  escreval("| L2 | ",sudoku[1,2]," | ",sudoku[2,2]," | ",sudoku[3,2]," |
",sudoku[4,2]," | ",sudoku[5,2]," | ",sudoku[6,2]," | ",sudoku[7,2]," |
",sudoku[8,2]," | ",sudoku[9,2]," |")
  escreval("|    | - - - - - | - - - - - | - - - - - |")
  escreval("| L3 | ",sudoku[1,3]," | ",sudoku[2,3]," | ",sudoku[3,3]," |
",sudoku[4,3]," | ",sudoku[5,3]," | ",sudoku[6,3]," | ",sudoku[7,3]," |
",sudoku[8,3]," | ",sudoku[9,3]," |")
  escreval("|    |-----------|-----------|-----------|")
  escreval("| L4 | ",sudoku[1,4]," | ",sudoku[2,4]," | ",sudoku[3,4]," |
",sudoku[4,4]," | ",sudoku[5,4]," | ",sudoku[6,4]," | ",sudoku[7,4]," |
",sudoku[8,4]," | ",sudoku[9,4]," |")
  escreval("|    | - - - - - | - - - - - | - - - - - |")
  escreval("| L5 | ",sudoku[1,5]," | ",sudoku[2,5]," | ",sudoku[3,5]," |
",sudoku[4,5]," | ",sudoku[5,5]," | ",sudoku[6,5]," | ",sudoku[7,5]," |
",sudoku[8,5]," | ",sudoku[9,5]," |")
  escreval("|    | - - - - - | - - - - - | - - - - - |")
  escreval("| L6 | ",sudoku[1,6]," | ",sudoku[2,6]," | ",sudoku[3,6]," |
",sudoku[4,6]," | ",sudoku[5,6]," | ",sudoku[6,6]," | ",sudoku[7,6]," |
",sudoku[8,6]," | ",sudoku[9,6]," |")
  escreval("|    |-----------|-----------|-----------|")
  escreval("| L7 | ",sudoku[1,7]," | ",sudoku[2,7]," | ",sudoku[3,7]," |
",sudoku[4,7]," | ",sudoku[5,7]," | ",sudoku[6,7]," | ",sudoku[7,7]," |
",sudoku[8,7]," | ",sudoku[9,7]," |")
  escreval("|    | - - - - - | - - - - - | - - - - - |")
```

```
  escreval("| L8 | ",sudoku[1,8]," | ",sudoku[2,8]," | ",sudoku[3,8]," |
",sudoku[4,8]," | ",sudoku[5,8]," | ",sudoku[6,8]," | ",sudoku[7,8]," |
",sudoku[8,8]," | ",sudoku[9,8]," |")
  escreval("|    | - - - - - - | - - - - - - | - - - - - - |")
  escreval("| L9 | ",sudoku[1,9]," | ",sudoku[2,9]," | ",sudoku[3,9]," |
",sudoku[4,9]," | ",sudoku[5,9]," | ",sudoku[6,9]," | ",sudoku[7,9]," |
",sudoku[8,9]," | ",sudoku[9,9]," |")
  escreval("| -- |-----------------------------------|")
  escreval("")

    // Digita 10 para retornar ao MENU PRINCIPAL.
  escreval("+----------------------------+")
  escreval("|    10 -> SAIR DO JOGO      |")
  escreval("+----------------------------+")
  // Exibe a pontuação do jogador segundo a dificuldade selecionada
  // e o número do cadastro.
  se w = 1 entao
   escreval("| PONTUAÇÃO ATUAL -> ", pontuser1[nuser1]:8,"|")
  senao

   se w = 2 entao
    escreval("| PONTUAÇÃO ATUAL -> ", pontuser2[nuser2]:8,"|")
   senao

    se w = 3 entao
     escreval("| PONTUAÇÃO ATUAL -> ", pontuser3[nuser3]:8,"|")
    fimse
   fimse
  fimse

  se l >= 81 entao
   // Caso a variável "l" atinga 81, aparece a mensagem de PARABÉNS!;
   // Variável "w" serve para determinar a dificuldade do jogo.
   escolha w
    caso 1
     escreval("+========================================+")
     escreval("|               PARABÉNS                 |")
     escreval("|         !!SUDOKU COMPLETADO!!          |")
     escreval("|                                        |")
     escreval("|JOGADOR ->> ", user1[nuser1]:29,"|")
     escreval("|PONTUAÇÃO FINAL ->> ",pontuser1[nuser1]:3,"
|")
     escreval("+========================================+")

    caso 2
     escreval("+========================================+")
     escreval("|               PARABÉNS                 |")
     escreval("|         !!SUDOKU COMPLETADO!!          |")
     escreval("|                                        |")
     escreval("|JOGADOR ->> ", user2[nuser2]:29,"|")
     escreval("|PONTUAÇÃO FINAL ->> ",pontuser2[nuser2]:3,"
|")
     escreval("+========================================+")
```

```
    caso 3
     escreval("+=========================================+")
     escreval("|                PARABÉNS                  |")
     escreval("|          !!SUDOKU COMPLETADO!!           |")
     escreval("|                                         |")
     escreval("|JOGADOR ->> ", user3[nuser3]:29,"|")
     escreval("|PONTUAÇÃO FINAL ->> ",pontuser3[nuser3]:3,"
|")
     escreval("+=========================================+")
   fimescolha

   escreval()
   escreval()
   // Aperta ENTER para retornar ao MENU PRINCIPAL.
   escreval("ENTER para retornar ao MENU PRINCIPAL")
   leia(k)
   limpatela
   menu

  senao
   escreval("+---------------------------+")
   escreval()
   // Insere a COLUNA onde está a casa que deseja completar;
   // Ou insira 10 para retornar ao MENU PRINCIPAL.
   escreva("COLUNA(C) ->> ")
   leia(a)
   // Enquanto a for menor que 1 ou maior que 10, pede para inserir
novamente.
   enquanto (a < 1) ou (a > 10) faca
    escreva("COLUNA(C) ->> ")
    leia(a)
   fimenquanto
   // Se "a" for igual 10, solicita confirmação para SAIR ou CONTINUAR.
   se a = 10 entao
    enquanto s <> 1 faca
     limpatela
     escreval("+---------------------------------+")
     escreval("| DESEJA MESMO SAIR DO JOGO ATUAL? |")
     escreval("| (todo o progresso será perdido)  |")
     escreval("|      1 -> SIM | 2 -> NÃO         |")
     escreval("+---------------------------------+")
     escreval()
     escreva("->> ")
     leia(s)
     se s = 2 entao // Continua o jogo.
      limpatela
      jogada
     fimse
    fimenquanto
    limpatela
    menu // Se "s" for igual a 1, então retorna ao MENU.
   fimse
```

```
   escreval()
   // Insere a LINHA onde está a casa que deseja completar.
   escreva("LINHA(L) ->> ")
   leia(b)
   // Enquanto "b" for menor que 1 ou maior que 9, pede para inserir
novamente.
   enquanto (b < 1) ou (b > 9) faca
    escreval()
    escreva("LINHA(L) ->> ")
    leia(b)
   fimenquanto
   escreval()
   escreval("COLUNA: ",a)
   escreval("LINHA: ",b)
   escreval()
   // Se a COLUNA e LINHA corresponderem a uma casa já preenchida,
   // o tabuleiro imprime novamente, e se inicia tendo que inserir
   // novamente a COLUNA e LINHA.
   se sudoku[a, b] <> " " entao
    escreval("ESTE ESPAÇO JÁ POSSUI UM NÚMERO!")
   senao
    // Senão, deve inserir um número para a casa correspondente.
    escreva("INSIRA O NÚMERO QUE DESEJA (1 - 9) ->> ")
    leia(c)
    // Enquanto "c" for diferente de 1, 2, 3, 4, 5, 6, 7, 8 e 9,
    // o programa pede para inserir um número válido.
    enquanto (c <> "1") e (c <> "2") e (c <> "3") e (c <> "4") e (c <> "5")
e (c <> "6") e (c <> "7") e (c <> "8") e (c <> "9") e (c <> "9") faca
     escreva("INSIRA O NÚMERO QUE DESEJA (1 - 9) ->> ")
     leia(c)
    fimenquanto

    // Se "c" for correspondente ao número certo para preencher o
tabuleiro,
    // é contado + 10 pontos para o jogador.
    se c = sudoku2[a,b] entao
     sudoku[a,b] <- c
     escreval("+ 10 PONTOS")
     se w = 1 entao
      pontuser1[nuser1] <- pontuser1[nuser1] + 10
     senao
      se w = 2 entao
       pontuser2[nuser2] <- pontuser2[nuser2] + 10
      senao
       se w = 3 entao
        pontuser3[nuser3] <- pontuser3[nuser3] + 10
       fimse
      fimse
     fimse
    senao
     // Senão, é descontado - 10 pontos do jogador;
     // É descontado somente se a pontuação do jogador
     // for maior ou igual a 10.
```

```
     se c <> sudoku2[a,b] entao
      se w = 1 entao
       se pontuser1[nuser1] >= 10 entao
        pontuser1[nuser1] <- pontuser1[nuser1] - 10
        escreval("- 10 PONTOS")
       fimse
      senao
       se w = 2 entao
        se pontuser2[nuser2] >= 10 entao
         pontuser2[nuser2] <- pontuser2[nuser2] - 10
         escreval("- 10 PONTOS")
        fimse
       senao
        se w = 3 entao
         se pontuser3[nuser3] >= 10 entao
          pontuser3[nuser3] <- pontuser3[nuser3] - 10
          escreval("- 10 PONTOS")
         fimse
        fimse
       fimse
      fimse
     fimse
    fimse
    escreval()
    // Aperte ENTER para retornar ao MENU PRINCIPAL.
    escreval(" ENTER para continuar")
    leia(k)
    limpatela
    escreval()
  fimse
 fimenquanto
 limpatela
 menu
fimprocedimento

// Início do Procedimento Determinar Dificuldade - "DIFICULDADES"
procedimento dificuldades
 Var
  vet1, vet2, vet3 : vetor[1..9,1..9] de caractere
  x,y,q,q1,q2,q3,n : inteiro
  vet4 : vetor[1..55,1..2] de inteiro
 inicio
  limpatela
  escreval("       ______  _________    ____  ")
  escreval("      / / __ \/ ____/   |  / __ \ ")
  escreval(" __  / / / / / / __/ /| | / /_/ / ")
  escreval("/ /_/ / /_/ / /_/ / ___ |/ _, _/  ")
  escreval("\____/\____/\____/_/  |_/_/ |_|   ")
  escreval()
  escreval("1 - FÁCIL")
  escreval("2 - MÉDIO")
  escreval("3 - DIFÍCIL")
```

```
  escreval()
  // Insere uma dificuldade disponível.
  escreva("INSIRA A DIFICULDADE DESEJADA ->> ")
  leia(w)
  // Enquanto for uma opção menor que 1 ou maior que 3,
  // solicita para inserir novamente.
  enquanto (w < 1) ou (w > 3) faca
   escreva("INSIRA UM DIFICULADADE VÁLIDA ->> ")
   leia(w)
  fimenquanto

  escolha w

   caso 1
    // DIFICULDADE FÁCIL
    cadastrof // Procedimento para cadastrar jogador na dificuldade FÁCIL
    aleatorio 1,3 // Escolhe um tabuleiro aleatoriamente da dificuldade
FÁCIL
    leia(x)
    aleatorio off

    escolha x

     caso 1
      // TABULEIRO 1;
      // Atribui os valores do tabuleiro do SUDOKU.
      vet1[1,1] <- "1"
      vet1[1,2] <- "7"
      vet1[1,3] <- "9"
      vet1[1,4] <- "6"
      vet1[1,5] <- "5"
      vet1[1,6] <- "2"
      vet1[1,7] <- "3"
      vet1[1,8] <- "4"
      vet1[1,9] <- "8"

      vet1[2,1] <- "8"
      vet1[2,2] <- "2"
      vet1[2,3] <- "6"
      vet1[2,4] <- "4"
      vet1[2,5] <- "9"
      vet1[2,6] <- "3"
      vet1[2,7] <- "1"
      vet1[2,8] <- "7"
      vet1[2,9] <- "5"

      vet1[3,1] <- "5"
      vet1[3,2] <- "4"
      vet1[3,3] <- "3"
      vet1[3,4] <- "1"
      vet1[3,5] <- "8"
      vet1[3,6] <- "7"
      vet1[3,7] <- "6"
```

```
vet1[3,8] <- "9"
vet1[3,9] <- "2"

vet1[4,1] <- "3"
vet1[4,2] <- "5"
vet1[4,3] <- "8"
vet1[4,4] <- "2"
vet1[4,5] <- "7"
vet1[4,6] <- "4"
vet1[4,7] <- "9"
vet1[4,8] <- "1"
vet1[4,9] <- "6"

vet1[5,1] <- "2"
vet1[5,2] <- "1"
vet1[5,3] <- "4"
vet1[5,4] <- "9"
vet1[5,5] <- "6"
vet1[5,6] <- "5"
vet1[5,7] <- "8"
vet1[5,8] <- "3"
vet1[5,9] <- "7"

vet1[6,1] <- "9"
vet1[6,2] <- "6"
vet1[6,3] <- "7"
vet1[6,4] <- "8"
vet1[6,5] <- "3"
vet1[6,6] <- "1"
vet1[6,7] <- "2"
vet1[6,8] <- "5"
vet1[6,9] <- "4"

vet1[7,1] <- "4"
vet1[7,2] <- "8"
vet1[7,3] <- "5"
vet1[7,4] <- "3"
vet1[7,5] <- "1"
vet1[7,6] <- "6"
vet1[7,7] <- "7"
vet1[7,8] <- "2"
vet1[7,9] <- "9"

vet1[8,1] <- "7"
vet1[8,2] <- "9"
vet1[8,3] <- "1"
vet1[8,4] <- "5"
vet1[8,5] <- "2"
vet1[8,6] <- "8"
vet1[8,7] <- "4"
vet1[8,8] <- "6"
vet1[8,9] <- "3"
```

```
vet1[9,1] <- "6"
vet1[9,2] <- "3"
vet1[9,3] <- "2"
vet1[9,4] <- "7"
vet1[9,5] <- "4"
vet1[9,6] <- "9"
vet1[9,7] <- "5"
vet1[9,8] <- "8"
vet1[9,9] <- "1"

// Atribui para cada posição da matriz "sudoku2"
// os números declarados na matriz "vet1".
para x de 1 ate 9 faca
 para y de 1 ate 9 faca
  sudoku2[x,y] <- vet1[x,y]
 fimpara
fimpara

// Apaga uma quantidade de números de forma aleatória do tabuleiro;
// A quantidade de números apagados depende da DIFICULDADE.
// Quanto mais difícil mais números apagados.
para q2 de 1 ate 40 faca
 // Escolhe dois números aleatoriamente de 1 - 9.
 aleatorio 1,9
 leia(q)
 leia(n)
 aleatorio off
 q1 <- 1

 // Testa se esses números são correspondentes
 // a um número já apagado no tabuleiro.
 enquanto (q1 >= 1) e (q1 <= 40) faca
  enquanto (q = vet4[q1,1]) e (n = vet4[q1,2]) faca
   aleatorio 1,9
   leia(q)
   leia(n)
   q1 <- 1
   aleatorio off
  fimenquanto
  q1 <- q1 + 1
 fimenquanto

 // Quando não for, outro número é apagado no tabuleiro
 // de forma aleatória.
 vet1[q,n] <- " "
 vet4[q2,1] <- q
 vet4[q2,2] <- n
fimpara
limpatela

// Cada elemento apagado é substituído
// do número original da matriz por " ".
para x de 1 ate 9 faca
```

```
  para y de 1 ate 9 faca
   sudoku[x,y] <- vet1[x,y]
  fimpara
 fimpara

caso 2
 // TABULEIRO 2;
 // Atribui os valores do tabuleiro do SUDOKU.
 vet1[1,1] <- "4"
 vet1[1,2] <- "5"
 vet1[1,3] <- "2"
 vet1[1,4] <- "1"
 vet1[1,5] <- "7"
 vet1[1,6] <- "3"
 vet1[1,7] <- "8"
 vet1[1,8] <- "9"
 vet1[1,9] <- "6"

 vet1[2,1] <- "1"
 vet1[2,2] <- "6"
 vet1[2,3] <- "8"
 vet1[2,4] <- "9"
 vet1[2,5] <- "2"
 vet1[2,6] <- "4"
 vet1[2,7] <- "5"
 vet1[2,8] <- "7"
 vet1[2,9] <- "3"

 vet1[3,1] <- "3"
 vet1[3,2] <- "7"
 vet1[3,3] <- "9"
 vet1[3,4] <- "5"
 vet1[3,5] <- "6"
 vet1[3,6] <- "8"
 vet1[3,7] <- "1"
 vet1[3,8] <- "2"
 vet1[3,9] <- "4"

 vet1[4,1] <- "8"
 vet1[4,2] <- "1"
 vet1[4,3] <- "7"
 vet1[4,4] <- "4"
 vet1[4,5] <- "9"
 vet1[4,6] <- "5"
 vet1[4,7] <- "6"
 vet1[4,8] <- "3"
 vet1[4,9] <- "2"

 vet1[5,1] <- "2"
 vet1[5,2] <- "4"
 vet1[5,3] <- "3"
 vet1[5,4] <- "6"
 vet1[5,5] <- "8"
```

```
vet1[5,6] <- "1"
vet1[5,7] <- "9"
vet1[5,8] <- "5"
vet1[5,9] <- "7"

vet1[6,1] <- "5"
vet1[6,2] <- "9"
vet1[6,3] <- "6"
vet1[6,4] <- "2"
vet1[6,5] <- "3"
vet1[6,6] <- "7"
vet1[6,7] <- "4"
vet1[6,8] <- "8"
vet1[6,9] <- "1"

vet1[7,1] <- "6"
vet1[7,2] <- "8"
vet1[7,3] <- "1"
vet1[7,4] <- "7"
vet1[7,5] <- "5"
vet1[7,6] <- "2"
vet1[7,7] <- "3"
vet1[7,8] <- "4"
vet1[7,9] <- "9"

vet1[8,1] <- "7"
vet1[8,2] <- "3"
vet1[8,3] <- "4"
vet1[8,4] <- "8"
vet1[8,5] <- "1"
vet1[8,6] <- "9"
vet1[8,7] <- "2"
vet1[8,8] <- "6"
vet1[8,9] <- "5"

vet1[9,1] <- "9"
vet1[9,2] <- "2"
vet1[9,3] <- "5"
vet1[9,4] <- "3"
vet1[9,5] <- "4"
vet1[9,6] <- "6"
vet1[9,7] <- "7"
vet1[9,8] <- "1"
vet1[9,9] <- "8"

// Atribui para cada posição da matriz "sudoku2"
// os números declarados na matriz "vet1".
para x de 1 ate 9 faca
 para y de 1 ate 9 faca
  sudoku2[x,y] <- vet1[x,y]
 fimpara
fimpara
```

```
 // Apaga uma quantidade de números de forma aleatória do tabuleiro;
 // A quantidade de números apagados depende da DIFICULDADE.
 // Quanto mais difícil mais números apagados.
 para q2 de 1 ate 40 faca
  // Escolhe dois números aleatoriamente de 1 - 9.
  aleatorio 1,9
  leia(q)
  leia(n)
  aleatorio off
  q1 <- 1

  // Testa se esses números são correspondentes
  // a um número já apagado no tabuleiro.
  enquanto (q1 >= 1) e (q1 <= 40) faca
   enquanto (q = vet4[q1,1]) e (n = vet4[q1,2]) faca
    aleatorio 1,9
    leia(q)
    leia(n)
    q1 <- 1
    aleatorio off
   fimenquanto
   q1 <- q1 + 1
  fimenquanto

  // Quando não for, outro número é apagado no tabuleiro
  // de forma aleatória.
  vet1[q,n] <- " "
  vet4[q2,1] <- q
  vet4[q2,2] <- n
 fimpara
 limpatela

 // Cada elemento apagado é substituído
 // do número original da matriz por " ".
 para x de 1 ate 9 faca
  para y de 1 ate 9 faca
   sudoku[x,y] <- vet1[x,y]
  fimpara
 fimpara

caso 3
 // TABULEIRO 3;
 // Atribui os valores do tabuleiro do SUDOKU.
 vet1[1,1] <- "8"
 vet1[1,2] <- "2"
 vet1[1,3] <- "4"
 vet1[1,4] <- "5"
 vet1[1,5] <- "1"
 vet1[1,6] <- "7"
 vet1[1,7] <- "6"
 vet1[1,8] <- "9"
 vet1[1,9] <- "3"
```

```
vet1[2,1] <- "3"
vet1[2,2] <- "9"
vet1[2,3] <- "1"
vet1[2,4] <- "6"
vet1[2,5] <- "2"
vet1[2,6] <- "4"
vet1[2,7] <- "5"
vet1[2,8] <- "8"
vet1[2,9] <- "7"

vet1[3,1] <- "5"
vet1[3,2] <- "6"
vet1[3,3] <- "7"
vet1[3,4] <- "9"
vet1[3,5] <- "3"
vet1[3,6] <- "8"
vet1[3,7] <- "2"
vet1[3,8] <- "1"
vet1[3,9] <- "4"

vet1[4,1] <- "4"
vet1[4,2] <- "8"
vet1[4,3] <- "2"
vet1[4,4] <- "1"
vet1[4,5] <- "6"
vet1[4,6] <- "5"
vet1[4,7] <- "7"
vet1[4,8] <- "3"
vet1[4,9] <- "9"

vet1[5,1] <- "1"
vet1[5,2] <- "5"
vet1[5,3] <- "9"
vet1[5,4] <- "3"
vet1[5,5] <- "7"
vet1[5,6] <- "2"
vet1[5,7] <- "8"
vet1[5,8] <- "4"
vet1[5,9] <- "6"

vet1[6,1] <- "6"
vet1[6,2] <- "7"
vet1[6,3] <- "3"
vet1[6,4] <- "4"
vet1[6,5] <- "8"
vet1[6,6] <- "9"
vet1[6,7] <- "1"
vet1[6,8] <- "5"
vet1[6,9] <- "2"

vet1[7,1] <- "9"
vet1[7,2] <- "4"
vet1[7,3] <- "6"
```

```
vet1[7,4] <- "7"
vet1[7,5] <- "5"
vet1[7,6] <- "1"
vet1[7,7] <- "3"
vet1[7,8] <- "2"
vet1[7,9] <- "8"

vet1[8,1] <- "2"
vet1[8,2] <- "3"
vet1[8,3] <- "5"
vet1[8,4] <- "8"
vet1[8,5] <- "4"
vet1[8,6] <- "6"
vet1[8,7] <- "9"
vet1[8,8] <- "7"
vet1[8,9] <- "1"

vet1[9,1] <- "7"
vet1[9,2] <- "1"
vet1[9,3] <- "8"
vet1[9,4] <- "2"
vet1[9,5] <- "9"
vet1[9,6] <- "3"
vet1[9,7] <- "4"
vet1[9,8] <- "6"
vet1[9,9] <- "5"

// Atribui para cada posição da matriz "sudoku2"
// os números declarados na matriz "vet1".
para x de 1 ate 9 faca
 para y de 1 ate 9 faca
  sudoku2[x,y] <- vet1[x,y]
 fimpara
fimpara

// Apaga uma quantidade de números de forma aleatória do tabuleiro;
// A quantidade de números apagados depende da DIFICULDADE.
// Quanto mais difícil mais números apagados.
para q2 de 1 ate 40 faca
 // Escolhe dois números aleatoriamente de 1 - 9.
 aleatorio 1,9
 leia(q)
 leia(n)
 aleatorio off
 q1 <- 1

 // Testa se esses números são correspondentes
 // a um número já apagado no tabuleiro.
 enquanto (q1 >= 1) e (q1 <= 40) faca
  enquanto (q = vet4[q1,1]) e (n = vet4[q1,2]) faca
   aleatorio 1,9
   leia(q)
   leia(n)
```

```
          q1 <- 1
          aleatorio off
         fimenquanto
         q1 <- q1 + 1
        fimenquanto

        // Quando não for, outro número é apagado no tabuleiro
        // de forma aleatória.
        vet1[q,n] <- " "
        vet4[q2,1] <- q
        vet4[q2,2] <- n
       fimpara
       limpatela

       // Cada elemento apagado é substituído
       // do número original da matriz por " ".
       para x de 1 ate 9 faca
        para y de 1 ate 9 faca
         sudoku[x,y] <- vet1[x,y]
        fimpara
       fimpara
      fimescolha

   caso 2
    // DIFICULDADE MÉDIA
    cadastrom // Procedimento para cadastrar jogador na dificuldade MÉDIA.
    aleatorio 1,3 // Escolhe um tabuleiro aleatoriamente da dificuldade
MÉDIA.
    leia(x)
    aleatorio off

    escolha x

     caso 1
      // TABULEIRO 1;
      // Atribui os valores do tabuleiro do SUDOKU.
      vet2[1,1] <- "5"
      vet2[1,2] <- "6"
      vet2[1,3] <- "1"
      vet2[1,4] <- "8"
      vet2[1,5] <- "4"
      vet2[1,6] <- "7"
      vet2[1,7] <- "9"
      vet2[1,8] <- "2"
      vet2[1,9] <- "3"

      vet2[2,1] <- "3"
      vet2[2,2] <- "7"
      vet2[2,3] <- "9"
      vet2[2,4] <- "5"
      vet2[2,5] <- "2"
      vet2[2,6] <- "1"
      vet2[2,7] <- "6"
```

```
vet2[2,8] <- "8"
vet2[2,9] <- "4"

vet2[3,1] <- "4"
vet2[3,2] <- "2"
vet2[3,3] <- "8"
vet2[3,4] <- "9"
vet2[3,5] <- "6"
vet2[3,6] <- "3"
vet2[3,7] <- "1"
vet2[3,8] <- "7"
vet2[3,9] <- "5"

vet2[4,1] <- "6"
vet2[4,2] <- "1"
vet2[4,3] <- "3"
vet2[4,4] <- "7"
vet2[4,5] <- "8"
vet2[4,6] <- "9"
vet2[4,7] <- "5"
vet2[4,8] <- "4"
vet2[4,9] <- "2"

vet2[5,1] <- "7"
vet2[5,2] <- "9"
vet2[5,3] <- "4"
vet2[5,4] <- "6"
vet2[5,5] <- "5"
vet2[5,6] <- "2"
vet2[5,7] <- "3"
vet2[5,8] <- "1"
vet2[5,9] <- "8"

vet2[6,1] <- "8"
vet2[6,2] <- "5"
vet2[6,3] <- "2"
vet2[6,4] <- "1"
vet2[6,5] <- "3"
vet2[6,6] <- "4"
vet2[6,7] <- "7"
vet2[6,8] <- "9"
vet2[6,9] <- "6"

vet2[7,1] <- "9"
vet2[7,2] <- "3"
vet2[7,3] <- "5"
vet2[7,4] <- "4"
vet2[7,5] <- "7"
vet2[7,6] <- "8"
vet2[7,7] <- "2"
vet2[7,8] <- "6"
vet2[7,9] <- "1"
```

```
vet2[8,1] <- "1"
vet2[8,2] <- "4"
vet2[8,3] <- "6"
vet2[8,4] <- "2"
vet2[8,5] <- "9"
vet2[8,6] <- "5"
vet2[8,7] <- "8"
vet2[8,8] <- "3"
vet2[8,9] <- "7"

vet2[9,1] <- "2"
vet2[9,2] <- "8"
vet2[9,3] <- "7"
vet2[9,4] <- "3"
vet2[9,5] <- "1"
vet2[9,6] <- "6"
vet2[9,7] <- "4"
vet2[9,8] <- "5"
vet2[9,9] <- "9"

// Atribui para cada posição da matriz "sudoku2"
// os números declarados na matriz "vet2".
para x de 1 ate 9 faca
 para y de 1 ate 9 faca
  sudoku2[x,y] <- vet2[x,y]
 fimpara
fimpara

// Apaga uma quantidade de números de forma aleatória do tabuleiro;
// A quantidade de números apagados depende da DIFICULDADE.
// Quanto mais difícil mais números apagados.
para q2 de 1 ate 50 faca
 // Escolhe dois números aleatoriamente de 1 - 9.
 aleatorio 1,9
 leia(q)
 leia(n)
 aleatorio off
 q1 <- 1

 // Testa se esses números são correspondentes
 // a um número já apagado no tabuleiro.
 enquanto (q1 >= 1) e (q1 <= 50) faca
  enquanto (q = vet4[q1,1]) e (n = vet4[q1,2]) faca
   aleatorio 1,9
   leia(q)
   leia(n)
   q1 <- 1
   aleatorio off
  fimenquanto
  q1 <- q1 + 1
 fimenquanto

 // Quando não for, outro número é apagado no tabuleiro
```

```
  // de forma aleatória.
  vet2[q,n] <- " "
  vet4[q2,1] <- q
  vet4[q2,2] <- n
 fimpara
 limpatela

 // Cada elemento apagado é substituído
 // do número original da matriz por " ".
 para x de 1 ate 9 faca
  para y de 1 ate 9 faca
   sudoku[x,y] <- vet2[x,y]
  fimpara
 fimpara

caso 2
 // TABULEIRO 2;
 // Atribui os valores do tabuleiro do SUDOKU.
 vet2[1,1] <- "5"
 vet2[1,2] <- "4"
 vet2[1,3] <- "6"
 vet2[1,4] <- "3"
 vet2[1,5] <- "8"
 vet2[1,6] <- "7"
 vet2[1,7] <- "2"
 vet2[1,8] <- "1"
 vet2[1,9] <- "9"

 vet2[2,1] <- "2"
 vet2[2,2] <- "9"
 vet2[2,3] <- "1"
 vet2[2,4] <- "6"
 vet2[2,5] <- "4"
 vet2[2,6] <- "5"
 vet2[2,7] <- "7"
 vet2[2,8] <- "8"
 vet2[2,9] <- "3"

 vet2[3,1] <- "8"
 vet2[3,2] <- "7"
 vet2[3,3] <- "3"
 vet2[3,4] <- "2"
 vet2[3,5] <- "1"
 vet2[3,6] <- "9"
 vet2[3,7] <- "6"
 vet2[3,8] <- "4"
 vet2[3,9] <- "5"

 vet2[4,1] <- "6"
 vet2[4,2] <- "3"
 vet2[4,3] <- "7"
 vet2[4,4] <- "5"
 vet2[4,5] <- "9"
```

```
vet2[4,6] <- "1"
vet2[4,7] <- "8"
vet2[4,8] <- "2"
vet2[4,9] <- "2"

vet2[5,1] <- "4"
vet2[5,2] <- "8"
vet2[5,3] <- "5"
vet2[5,4] <- "7"
vet2[5,5] <- "3"
vet2[5,6] <- "2"
vet2[5,7] <- "1"
vet2[5,8] <- "9"
vet2[5,9] <- "6"

vet2[6,1] <- "9"
vet2[6,2] <- "1"
vet2[6,3] <- "2"
vet2[6,4] <- "4"
vet2[6,5] <- "6"
vet2[6,6] <- "8"
vet2[6,7] <- "3"
vet2[6,8] <- "5"
vet2[6,9] <- "7"

vet2[7,1] <- "3"
vet2[7,2] <- "2"
vet2[7,3] <- "9"
vet2[7,4] <- "1"
vet2[7,5] <- "7"
vet2[7,6] <- "4"
vet2[7,7] <- "5"
vet2[7,8] <- "6"
vet2[7,9] <- "8"

vet2[8,1] <- "7"
vet2[8,2] <- "5"
vet2[8,3] <- "8"
vet2[8,4] <- "9"
vet2[8,5] <- "2"
vet2[8,6] <- "6"
vet2[8,7] <- "4"
vet2[8,8] <- "3"
vet2[8,9] <- "1"

vet2[9,1] <- "1"
vet2[9,2] <- "6"
vet2[9,3] <- "4"
vet2[9,4] <- "8"
vet2[9,5] <- "5"
vet2[9,6] <- "3"
vet2[9,7] <- "9"
vet2[9,8] <- "7"
```

```
 vet2[9,9] <- "2"

 // Atribui para cada posição da matriz "sudoku2"
 // os números declarados na matriz "vet2".
 para x de 1 ate 9 faca
  para y de 1 ate 9 faca
   sudoku2[x,y] <- vet2[x,y]
  fimpara
 fimpara

 // Apaga uma quantidade de números de forma aleatória do tabuleiro;
 // A quantidade de números apagados depende da DIFICULDADE.
 // Quanto mais difícil mais números apagados.
 para q2 de 1 ate 50 faca
  // Escolhe dois números aleatoriamente de 1 - 9.
  aleatorio 1,9
  leia(q)
  leia(n)
  aleatorio off
  q1 <- 1

  // Testa se esses números são correspondentes
  // a um número já apagado no tabuleiro.
  enquanto (q1 >= 1) e (q1 <= 50) faca
   enquanto (q = vet4[q1,1]) e (n = vet4[q1,2]) faca
    aleatorio 1,9
    leia(q)
    leia(n)
    q1 <- 1
    aleatorio off
   fimenquanto
   q1 <- q1 + 1
  fimenquanto

  // Quando não for, outro número é apagado no tabuleiro
  // de forma aleatória.
  vet2[q,n] <- " "
  vet4[q2,1] <- q
  vet4[q2,2] <- n
 fimpara
 limpatela

 // Cada elemento apagado é substituído
 // do número original da matriz por " ".
 para x de 1 ate 9 faca
  para y de 1 ate 9 faca
   sudoku[x,y] <- vet2[x,y]
  fimpara
 fimpara

caso 3
 // TABULEIRO 3;
 // Atribui os valores do tabuleiro do SUDOKU.
```

```
vet2[1,1] <- "1"
vet2[1,2] <- "9"
vet2[1,3] <- "8"
vet2[1,4] <- "7"
vet2[1,5] <- "6"
vet2[1,6] <- "4"
vet2[1,7] <- "5"
vet2[1,8] <- "2"
vet2[1,9] <- "3"

vet2[2,1] <- "3"
vet2[2,2] <- "2"
vet2[2,3] <- "6"
vet2[2,4] <- "5"
vet2[2,5] <- "1"
vet2[2,6] <- "8"
vet2[2,7] <- "7"
vet2[2,8] <- "9"
vet2[2,9] <- "4"

vet2[3,1] <- "7"
vet2[3,2] <- "5"
vet2[3,3] <- "4"
vet2[3,4] <- "3"
vet2[3,5] <- "2"
vet2[3,6] <- "9"
vet2[3,7] <- "1"
vet2[3,8] <- "8"
vet2[3,9] <- "6"

vet2[4,1] <- "9"
vet2[4,2] <- "3"
vet2[4,3] <- "5"
vet2[4,4] <- "8"
vet2[4,5] <- "7"
vet2[4,6] <- "6"
vet2[4,7] <- "4"
vet2[4,8] <- "1"
vet2[4,9] <- "2"

vet2[5,1] <- "8"
vet2[5,2] <- "4"
vet2[5,3] <- "2"
vet2[5,4] <- "1"
vet2[5,5] <- "3"
vet2[5,6] <- "5"
vet2[5,7] <- "9"
vet2[5,8] <- "6"
vet2[5,9] <- "7"

vet2[6,1] <- "6"
vet2[6,2] <- "7"
vet2[6,3] <- "1"
```

```
vet2[6,4] <- "4"
vet2[6,5] <- "9"
vet2[6,6] <- "2"
vet2[6,7] <- "3"
vet2[6,8] <- "5"
vet2[6,9] <- "8"

vet2[7,1] <- "4"
vet2[7,2] <- "1"
vet2[7,3] <- "9"
vet2[7,4] <- "6"
vet2[7,5] <- "8"
vet2[7,6] <- "3"
vet2[7,7] <- "2"
vet2[7,8] <- "7"
vet2[7,9] <- "5"

vet2[8,1] <- "5"
vet2[8,2] <- "6"
vet2[8,3] <- "7"
vet2[8,4] <- "2"
vet2[8,5] <- "4"
vet2[8,6] <- "1"
vet2[8,7] <- "8"
vet2[8,8] <- "3"
vet2[8,9] <- "9"

vet2[9,1] <- "2"
vet2[9,2] <- "8"
vet2[9,3] <- "3"
vet2[9,4] <- "9"
vet2[9,5] <- "5"
vet2[9,6] <- "7"
vet2[9,7] <- "6"
vet2[9,8] <- "4"
vet2[9,9] <- "1"

// Atribui para cada posição da matriz "sudoku2"
// os números declarados na matriz "vet2".
para x de 1 ate 9 faca
 para y de 1 ate 9 faca
  sudoku2[x,y] <- vet2[x,y]
 fimpara
fimpara

// Apaga uma quantidade de números de forma aleatória do tabuleiro;
// A quantidade de números apagados depende da DIFICULDADE.
// Quanto mais difícil mais números apagados.
para q2 de 1 ate 50 faca
 // Escolhe dois números aleatoriamente de 1 - 9.
 aleatorio 1,9
 leia(q)
 leia(n)
```

```
        aleatorio off
        q1 <- 1

        // Testa se esses números são correspondentes
        // a um número já apagado no tabuleiro.
        enquanto (q1 >= 1) e (q1 <= 50) faca
         enquanto (q = vet4[q1,1]) e (n = vet4[q1,2]) faca
          aleatorio 1,9
          leia(q)
          leia(n)
          q1 <- 1
          aleatorio off
         fimenquanto
         q1 <- q1 + 1
        fimenquanto

        // Quando não for, outro número é apagado no tabuleiro
        // de forma aleatória.
        vet2[q,n] <- " "
        vet4[q2,1] <- q
        vet4[q2,2] <- n
       fimpara
       limpatela

       // Cada elemento apagado é substituído
       // do número original da matriz por " ".
       para x de 1 ate 9 faca
        para y de 1 ate 9 faca
         sudoku[x,y] <- vet2[x,y]
        fimpara
       fimpara

      fimescolha


   caso 3
    // DIFICULDADE DIFÍCIL
    cadastrod // Procedimento para cadastrar jogador na dificuldade
DIFÍCIL.
    aleatorio 1,3 // Escolhe um tabuleiro aleatoriamente da dificuldade
DIFÍCIL.
    leia(x)
    aleatorio off

    escolha x

     caso 1
      // TABULEIRO 1;
      // Atribui os valores do tabuleiro do SUDOKU.
      vet3[1,1] <- "6"
      vet3[1,2] <- "2"
      vet3[1,3] <- "3"
      vet3[1,4] <- "1"
```

```
vet3[1,5] <- "7"
vet3[1,6] <- "4"
vet3[1,7] <- "8"
vet3[1,8] <- "9"
vet3[1,9] <- "5"

vet3[2,1] <- "5"
vet3[2,2] <- "4"
vet3[2,3] <- "9"
vet3[2,4] <- "3"
vet3[2,5] <- "2"
vet3[2,6] <- "8"
vet3[2,7] <- "6"
vet3[2,8] <- "1"
vet3[2,9] <- "7"

vet3[3,1] <- "7"
vet3[3,2] <- "8"
vet3[3,3] <- "1"
vet3[3,4] <- "6"
vet3[3,5] <- "5"
vet3[3,6] <- "9"
vet3[3,7] <- "3"
vet3[3,8] <- "4"
vet3[3,9] <- "2"

vet3[4,1] <- "3"
vet3[4,2] <- "9"
vet3[4,3] <- "5"
vet3[4,4] <- "8"
vet3[4,5] <- "1"
vet3[4,6] <- "7"
vet3[4,7] <- "2"
vet3[4,8] <- "6"
vet3[4,9] <- "4"

vet3[5,1] <- "2"
vet3[5,2] <- "1"
vet3[5,3] <- "4"
vet3[5,4] <- "5"
vet3[5,5] <- "9"
vet3[5,6] <- "6"
vet3[5,7] <- "7"
vet3[5,8] <- "3"
vet3[5,9] <- "8"

vet3[6,1] <- "8"
vet3[6,2] <- "7"
vet3[6,3] <- "6"
vet3[6,4] <- "2"
vet3[6,5] <- "4"
vet3[6,6] <- "3"
vet3[6,7] <- "9"
```

```
vet3[6,8] <- "5"
vet3[6,9] <- "1"

vet3[7,1] <- "4"
vet3[7,2] <- "3"
vet3[7,3] <- "2"
vet3[7,4] <- "7"
vet3[7,5] <- "6"
vet3[7,6] <- "1"
vet3[7,7] <- "5"
vet3[7,8] <- "8"
vet3[7,9] <- "9"

vet3[8,1] <- "1"
vet3[8,2] <- "6"
vet3[8,3] <- "7"
vet3[8,4] <- "9"
vet3[8,5] <- "8"
vet3[8,6] <- "5"
vet3[8,7] <- "4"
vet3[8,8] <- "2"
vet3[8,9] <- "3"

vet3[9,1] <- "9"
vet3[9,2] <- "5"
vet3[9,3] <- "8"
vet3[9,4] <- "4"
vet3[9,5] <- "3"
vet3[9,6] <- "2"
vet3[9,7] <- "1"
vet3[9,8] <- "7"
vet3[9,9] <- "6"

// Atribui para cada posição da matriz "sudoku2"
// os números declarados na matriz "vet3".
para x de 1 ate 9 faca
 para y de 1 ate 9 faca
  sudoku2[x,y] <- vet3[x,y]
 fimpara
fimpara

// Apaga uma quantidade de números de forma aleatória do tabuleiro;
// A quantidade de números apagados depende da DIFICULDADE.
// Quanto mais difícil mais números apagados.
para q2 de 1 ate 55 faca
 // Escolhe dois números aleatoriamente de 1 - 9.
 aleatorio 1,9
 leia(q)
 leia(n)
 aleatorio off
 q1 <- 1

 // Testa se esses números são correspondentes
```

```
  // a um número já apagado no tabuleiro.
  enquanto (q1 >= 1) e (q1 <= 55) faca
   enquanto (q = vet4[q1,1]) e (n = vet4[q1,2]) faca
    aleatorio 1,9
    leia(q)
    leia(n)
    q1 <- 1
    aleatorio off
   fimenquanto
   q1 <- q1 + 1
  fimenquanto

  // Quando não for, outro número é apagado no tabuleiro
  // de forma aleatória.
  vet3[q,n] <- " "
  vet4[q2,1] <- q
  vet4[q2,2] <- n
 fimpara
 limpatela

 // Cada elemento apagado é substituído
 // do número original da matriz por " ".
 para x de 1 ate 9 faca
  para y de 1 ate 9 faca
   sudoku[x,y] <- vet3[x,y]
  fimpara
 fimpara

caso 2
 // TABULEIRO 2;
 // Atribui os valores do tabuleiro do SUDOKU.
 vet3[1,1] <- "5"
 vet3[1,2] <- "6"
 vet3[1,3] <- "7"
 vet3[1,4] <- "1"
 vet3[1,5] <- "2"
 vet3[1,6] <- "8"
 vet3[1,7] <- "3"
 vet3[1,8] <- "9"
 vet3[1,9] <- "4"

 vet3[2,1] <- "9"
 vet3[2,2] <- "8"
 vet3[2,3] <- "4"
 vet3[2,4] <- "3"
 vet3[2,5] <- "5"
 vet3[2,6] <- "7"
 vet3[2,7] <- "2"
 vet3[2,8] <- "6"
 vet3[2,9] <- "1"

 vet3[3,1] <- "3"
 vet3[3,2] <- "1"
```

```
vet3[3,3] <- "2"
vet3[3,4] <- "4"
vet3[3,5] <- "6"
vet3[3,6] <- "9"
vet3[3,7] <- "8"
vet3[3,8] <- "5"
vet3[3,9] <- "7"

vet3[4,1] <- "4"
vet3[4,2] <- "7"
vet3[4,3] <- "8"
vet3[4,4] <- "5"
vet3[4,5] <- "1"
vet3[4,6] <- "2"
vet3[4,7] <- "6"
vet3[4,8] <- "3"
vet3[4,9] <- "9"

vet3[5,1] <- "1"
vet3[5,2] <- "2"
vet3[5,3] <- "3"
vet3[5,4] <- "9"
vet3[5,5] <- "4"
vet3[5,6] <- "6"
vet3[5,7] <- "5"
vet3[5,8] <- "7"
vet3[5,9] <- "8"

vet3[6,1] <- "6"
vet3[6,2] <- "9"
vet3[6,3] <- "5"
vet3[6,4] <- "7"
vet3[6,5] <- "8"
vet3[6,6] <- "3"
vet3[6,7] <- "1"
vet3[6,8] <- "4"
vet3[6,9] <- "2"

vet3[7,1] <- "8"
vet3[7,2] <- "3"
vet3[7,3] <- "9"
vet3[7,4] <- "6"
vet3[7,5] <- "7"
vet3[7,6] <- "1"
vet3[7,7] <- "4"
vet3[7,8] <- "2"
vet3[7,9] <- "5"

vet3[8,1] <- "7"
vet3[8,2] <- "4"
vet3[8,3] <- "1"
vet3[8,4] <- "2"
vet3[8,5] <- "3"
```

```
vet3[8,6] <- "5"
vet3[8,7] <- "9"
vet3[8,8] <- "8"
vet3[8,9] <- "6"

vet3[9,1] <- "2"
vet3[9,2] <- "5"
vet3[9,3] <- "6"
vet3[9,4] <- "8"
vet3[9,5] <- "9"
vet3[9,6] <- "4"
vet3[9,7] <- "7"
vet3[9,8] <- "1"
vet3[9,9] <- "3"

// Atribui para cada posição da matriz "sudoku2"
// os números declarados na matriz "vet3".
para x de 1 ate 9 faca
 para y de 1 ate 9 faca
  sudoku2[x,y] <- vet3[x,y]
 fimpara
fimpara

// Apaga uma quantidade de números de forma aleatória do tabuleiro;
// A quantidade de números apagados depende da DIFICULDADE.
// Quanto mais difícil mais números apagados.
para q2 de 1 ate 55 faca
 // Escolhe dois números aleatoriamente de 1 - 9.
 aleatorio 1,9
 leia(q)
 leia(n)
 aleatorio off
 q1 <- 1

 // Testa se esses números são correspondentes
 // a um número já apagado no tabuleiro.
 enquanto (q1 >= 1) e (q1 <= 55) faca
  enquanto (q = vet4[q1,1]) e (n = vet4[q1,2]) faca
   aleatorio 1,9
   leia(q)
   leia(n)
   q1 <- 1
   aleatorio off
  fimenquanto
  q1 <- q1 + 1
 fimenquanto

 // Quando não for, outro número é apagado no tabuleiro
 // de forma aleatória.
 vet3[q,n] <- " "
 vet4[q2,1] <- q
 vet4[q2,2] <- n
fimpara
```

```
 limpatela

 // Cada elemento apagado é substituído
 // do número original da matriz por " ".
 para x de 1 ate 9 faca
  para y de 1 ate 9 faca
   sudoku[x,y] <- vet3[x,y]
  fimpara
 fimpara

caso 3
 // TABULEIRO 3;
 // Atribui os valores do tabuleiro do SUDOKU.
 vet3[1,1] <- "4"
 vet3[1,2] <- "7"
 vet3[1,3] <- "3"
 vet3[1,4] <- "5"
 vet3[1,5] <- "9"
 vet3[1,6] <- "6"
 vet3[1,7] <- "8"
 vet3[1,8] <- "1"
 vet3[1,9] <- "2"

 vet3[2,1] <- "8"
 vet3[2,2] <- "6"
 vet3[2,3] <- "9"
 vet3[2,4] <- "2"
 vet3[2,5] <- "1"
 vet3[2,6] <- "3"
 vet3[2,7] <- "4"
 vet3[2,8] <- "7"
 vet3[2,9] <- "5"

 vet3[3,1] <- "2"
 vet3[3,2] <- "1"
 vet3[3,3] <- "5"
 vet3[3,4] <- "7"
 vet3[3,5] <- "4"
 vet3[3,6] <- "8"
 vet3[3,7] <- "9"
 vet3[3,8] <- "3"
 vet3[3,9] <- "6"

 vet3[4,1] <- "5"
 vet3[4,2] <- "2"
 vet3[4,3] <- "6"
 vet3[4,4] <- "4"
 vet3[4,5] <- "8"
 vet3[4,6] <- "1"
 vet3[4,7] <- "7"
 vet3[4,8] <- "9"
 vet3[4,9] <- "3"
```

```
vet3[5,1] <- "1"
vet3[5,2] <- "3"
vet3[5,3] <- "7"
vet3[5,4] <- "9"
vet3[5,5] <- "6"
vet3[5,6] <- "2"
vet3[5,7] <- "5"
vet3[5,8] <- "4"
vet3[5,9] <- "8"

vet3[6,1] <- "9"
vet3[6,2] <- "8"
vet3[6,3] <- "4"
vet3[6,4] <- "3"
vet3[6,5] <- "7"
vet3[6,6] <- "5"
vet3[6,7] <- "6"
vet3[6,8] <- "2"
vet3[6,9] <- "1"

vet3[7,1] <- "3"
vet3[7,2] <- "4"
vet3[7,3] <- "2"
vet3[7,4] <- "6"
vet3[7,5] <- "5"
vet3[7,6] <- "7"
vet3[7,7] <- "1"
vet3[7,8] <- "8"
vet3[7,9] <- "9"

vet3[8,1] <- "7"
vet3[8,2] <- "5"
vet3[8,3] <- "1"
vet3[8,4] <- "8"
vet3[8,5] <- "3"
vet3[8,6] <- "9"
vet3[8,7] <- "2"
vet3[8,8] <- "6"
vet3[8,9] <- "4"

vet3[9,1] <- "6"
vet3[9,2] <- "9"
vet3[9,3] <- "8"
vet3[9,4] <- "1"
vet3[9,5] <- "2"
vet3[9,6] <- "4"
vet3[9,7] <- "3"
vet3[9,8] <- "5"
vet3[9,9] <- "7"

// Atribui para cada posição da matriz "sudoku2"
// os números declarados na matriz "vet3".
para x de 1 ate 9 faca
```

```
        para y de 1 ate 9 faca
         sudoku2[x,y] <- vet3[x,y]
        fimpara
       fimpara

       // Apaga uma quantidade de números de forma aleatória do tabuleiro;
       // A quantidade de números apagados depende da DIFICULDADE.
       // Quanto mais difícil mais números apagados.
       para q2 de 1 ate 55 faca
        aleatorio 1,9
        leia(q)
        leia(n)
        aleatorio off
        q1 <- 1

        // Testa se esses números são correspondentes
        // a um número já apagado no tabuleiro.
        enquanto (q1 >= 1) e (q1 <= 55) faca
         enquanto (q = vet4[q1,1]) e (n = vet4[q1,2]) faca
          aleatorio 1,9
          leia(q)
          leia(n)
          q1 <- 1
          aleatorio off
         fimenquanto
         q1 <- q1 + 1
        fimenquanto

        // Quando não for, outro número é apagado no tabuleiro
        // de forma aleatória.
        vet3[q,n] <- " "
        vet4[q2,1] <- q
        vet4[q2,2] <- n
       fimpara
       limpatela

       // Cada elemento apagado é substituído
       // do número original da matriz por " ".
       para x de 1 ate 9 faca
        para y de 1 ate 9 faca
         sudoku[x,y] <- vet3[x,y]
        fimpara
       fimpara

   fimescolha
  fimescolha
fimprocedimento

 //INÍCIO DO PROGRAMA
 // O Algoritmo funciona com base em procedimentos
 // estando todos juntos no procedimento "MENU";
 // Com excessão da mensagem de "BEM VINDO" que inicia o algoritmo.
```

```
 //BANNER JOGO
timer(30)
 limpatela
 escreval("
_____________________________________________________ ")
 escreval("                              ########  ######## ##     ##
")
 escreval("                              ##     ## ##       ###   ###
")
 escreval("                              ##     ## ##       #### ####
")
 escreval("                              ########  ######   ## ### ##
")
 escreval("                              ##     ## ##       ##     ##
")
 escreval("                              ##     ## ##       ##     ##
")
 escreval("                              ########  ######## ##     ##
")
 escreval("
_____________________________________________________ ")
 escreval("
_____________________________________________________ ")
 escreval("                        ##     ## #### ##    ## ########   #######
")
 escreval("                        ##     ##  ##  ###   ## ##     ## ##     ##
")
 escreval("                        ##     ##  ##  ####  ## ##     ## ##     ##
")
 escreval("                        ##     ##  ##  ## ## ## ##     ## ##     ##
")
 escreval("                         ##   ##   ##  ##  #### ##     ## ##     ##
")
 escreval("                          ## ##    ##  ##   ### ##     ## ##     ##
")
 escreval("                           ###    #### ##    ## ########   #######
")
 escreval("
_____________________________________________________ ")
 escreval("
_____________________________________________________ ")
 escreval("                                 ###     #######
")
 escreval("                                ## ##   ##     ##
")
 escreval("                               ##   ##  ##     ##
")
 escreval("                              ##     ## ##     ##
")
 escreval("                              ######### ##     ##
")
 escreval("                              ##     ## ##     ##
")
```

```
 escreval("                                ##     ##  #######
")
 escreval("
_____________________________________________________ ")
 escreval("
_____________________________________________________")
 escreval("
")
 escreval("                  ####### ##     ## #####     #######  ##   ## ##
## ")
 escreval("                  ##      ##     ## ##  ##   ##     ## ##  ##  ##
## ")
 escreval("                  ##      ##     ## ##   ##  ##     ## ## ##   ##
## ")
 escreval("                  ####### ##     ## ##    ## ##     ## ####    ##
## ")
 escreval("                       ## ##     ## ##    ## ##     ## ####    ##
## ")
 escreval("                       ## ##     ## ##   ##  ##     ## ## ##   ##
## ")
 escreval("                  #######  #######  ######    #######  ##  ##
#######  ")
 escreval("
_____________________________________________________")
timer(0)
escreval()
escreval()
escreval("                          PRESSIONE QUALQUER BOTÃO PARA COMEÇAR")
escreval()
leia(k)
limpatela

menu


fimalgoritmo
```