**Samuel Rodrigues Viana de Faria**

**Thomas Santos Pollarini**

**Relatório sobre o Código de Resolução de Sudoku**

**1. Problema Abordado:**

O código em questão aborda o problema da resolução de um jogo de Sudoku. O Sudoku é um quebra-cabeça numérico em que o objetivo é preencher uma grade 9x9 com números de 1 a 9, de modo que cada linha, cada coluna e cada um dos nove subgrids 3x3 (chamados de "quadrantes") contenham todos os números de 1 a 9, sem repetições.

**2. Modelagem do Problema:**

O código é dividido em várias funções:

a) **gerar\_sudoku(sudoku\_str):** Esta função recebe uma representação de string do tabuleiro de Sudoku, onde os números são separados por espaços e as linhas são separadas por quebras de linha. A função valida a entrada, garantindo que haja 9 linhas com 9 números cada e converte a representação em uma matriz bidimensional.

b) **ler\_arquivo\_para\_string(nome\_arquivo):** Esta função lê um arquivo de texto contendo o tabuleiro de Sudoku e retorna seu conteúdo como uma string.

c) **verificaColocaNum(num, lin, col):** Esta função verifica se é possível colocar um número em uma célula específica do tabuleiro, seguindo as regras do Sudoku. Ela verifica se o número já existe na mesma linha, coluna ou no quadrante 3x3 correspondente.

d) **resolve():** Esta é a função principal responsável pela resolução do Sudoku usando uma abordagem de força bruta com recursão. Ela verifica se um número pode ser colocado em uma célula do tabuleiro, preenche as células vazias e verifica se o tabuleiro está completo e correto. Em casos onde o caminho tomado para a solução não seja possível, a função realiza o backtracking para verificar outras soluções possíveis.

e) **resolverSudoku(matrizSudoku):** Esta função chama a função resolve() e imprime a solução do Sudoku ou uma mensagem indicando que não há solução.

**3. Pontos Fortes:**

Os pontos fortes do código incluem a capacidade de resolver a maioria dos tabuleiros válidos de Sudoku, uso eficaz de recursão para exploração de possibilidades e flexibilidade ao permitir ao usuário inserir o caminho do arquivo de entrada.

**4. Pontos Fracos:**

Os pontos fracos do código de resolução de Sudoku incluem a abordagem de força bruta que pode ser lenta para tabuleiros complexos, a falta de validação completa da entrada, ausência de tratamento de exceções para casos insolúveis e uma saída limitada que não fornece detalhes sobre o processo de resolução.

**5. Conclusão:**

O código fornece uma implementação funcional para resolver tabuleiros de Sudoku usando backtracking. Ele é eficaz para tabuleiros válidos e relativamente simples. No entanto, carece de otimização para lidar com tabuleiros mais complexos. Melhorias adicionais poderiam ser feitas para tornar o código mais eficiente e robusto.