

**Projeto Sudoku**

**Jogo desenvolvido em C**

**Autores:**Laís de Sousa Barreto

Yuri Marconi Gouveia Ribeiro

Jessé do Nascimento Alves

João pessoa

2020

**Autores:**Laís de Sousa Barreto

Yuri Marconi Gouveia Ribeiro

Jessé do Nascimento Alves

**Projeto Sudoku**

**Jogo desenvolvido em C**

Relatório pertencente à matéria Técnicas de Desenvolvimento de Algoritmos, referente ao projeto do jogo Sudoku desenvolvido na linguagem C, componente da segunda nota.

Prof. Leandro Figueiredo Alves

João pessoa

2020

**Resumo**

A ideia do jogo Sudoku digital foi concebida por nosso grupo por se tratar de um jogo divertido que envolve muito raciocínio lógico. Implementamos na linguagem C, usando várias funções e interface de usuário pelo terminal. Possui duas opções para cada nível, exceto o difícil, que é um desafio único (e muito difícil!). Além disso, conta ainda com um sistema exclusivo de pontuação, em que não apenas a jogada certa influencia a pontuação final, mas o tempo decorrido ao total também.

**1 - Introdução**

O Sudoku é um jogo mundialmente conhecido, inventado por Howard Garns, arquiteto aposentado de Nova York que gostava muito de quebra-cabeças. Na verdade, o nome original é *Number Place* (ainda chamado assim nos Estados Unidos), traduzido literalmente como Lugar do Número, porém se popularizou posteriormente como Sudoku, abreviação da frase japonesa que significa “os dígitos devem permanecer únicos”.

Sudoku funciona assim: em uma tabela 9x9, com tabelas internas 3x3, o jogo inicia com alguns números em certos lugares, e o objetivo é preencher as lacunas com números que não se repitam tantoo nas tabelas 3x3, como nas linhas e colunas, indo de 1 a 9.

Normalmente, o Sudoku é classificado de acordo com níveis de dificuldade, que influenciam na estratégia adotada para resolução do jogo. Quanto mais difícil, menos números são oferecidos no inicío como ponto de partida, e mais complexa a análise e estratégia necessárias para fechar a tabela.

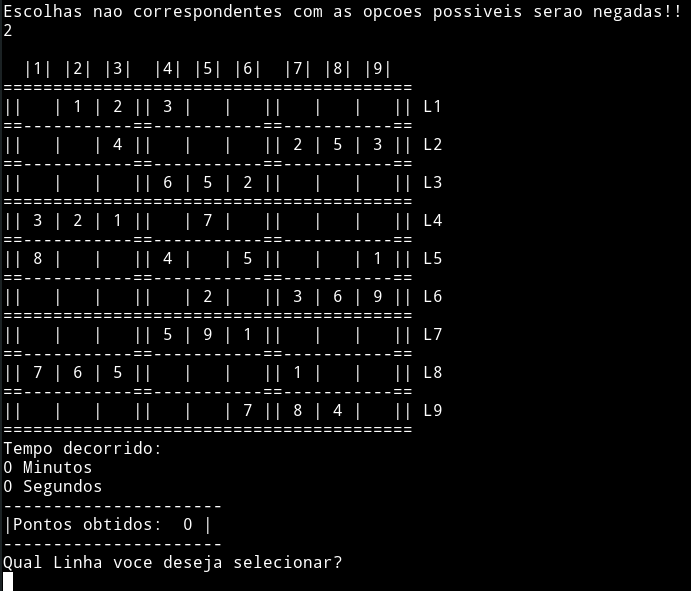
**2 - Execução do Algoritmo**

**2.1 - Tela de apresentação do jogo**



Nesta primeira tela, é fornecido um menu de opções de nível para que o usuário escolha o que deseja resolver. A opção deve coincidir com o número no menu, se não será ignorado e terá de escolher novamente.

**2.2 - Tabela do jogo e jogada**



Ao escolher o nível fácil através da opção 2, por exemplo, esta tela aparece, com a tabela 9x9 a ser preenchida pelo terminal utilizando a linha, a coluna e por fim o número escolhido pela pessoa para preencher a lacuna. Mostra também o tempo decorrido e os pontos obtidos após cada jogada.

Observe a imagem abaixo que exemplifica isso:



Como pode ver, decorreram-se 3 minutos e 22 segundos desde o início do jogo e a última partida, e foram obtidos 2 pontos ao acertar, preenchendo a lacuna com o número certo e recarregando a tabela com o número no lugar. Se errar, serão descontados 4 pontos.

Observe abaixo:



Se colocar número cuja coordenada já foi preenchida, mostra uma mensagem de aviso “coordenada invalida ou ja possui numero!!”.

Observe abaixo:



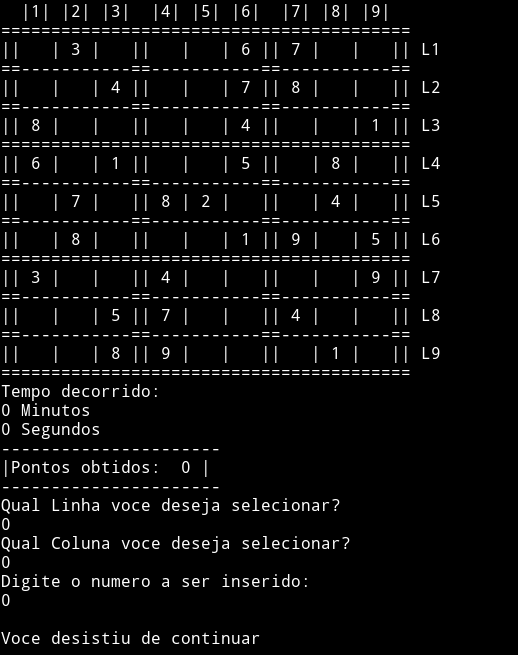
**2.3 - Resolução do Jogo**



Esta é a tela que aparece ao vencer o jogo. Note que há um bônus obtido em virtude do tempo. No caso, 200 pontos por terminar antes de 20 minutos. Entre 20 e 30 minutos, seria 100 pontos. Antes de 5 minutos, 400 pontos, e entre 5 e 10 minutos, 300 pontos. Após 30 minutos, não há bônus.

Decorridos 5 segundos, a tela sobe, dando lugar ao menu do jogo novamente.

Para sair do jogo, selecione 0. Se já iniciou a partida, digite 0 na opção da linha, da coluna e do número a preencher, mostrando a seguir a mensagem “Voce desistiu de continuar”, demonstrado a seguir:

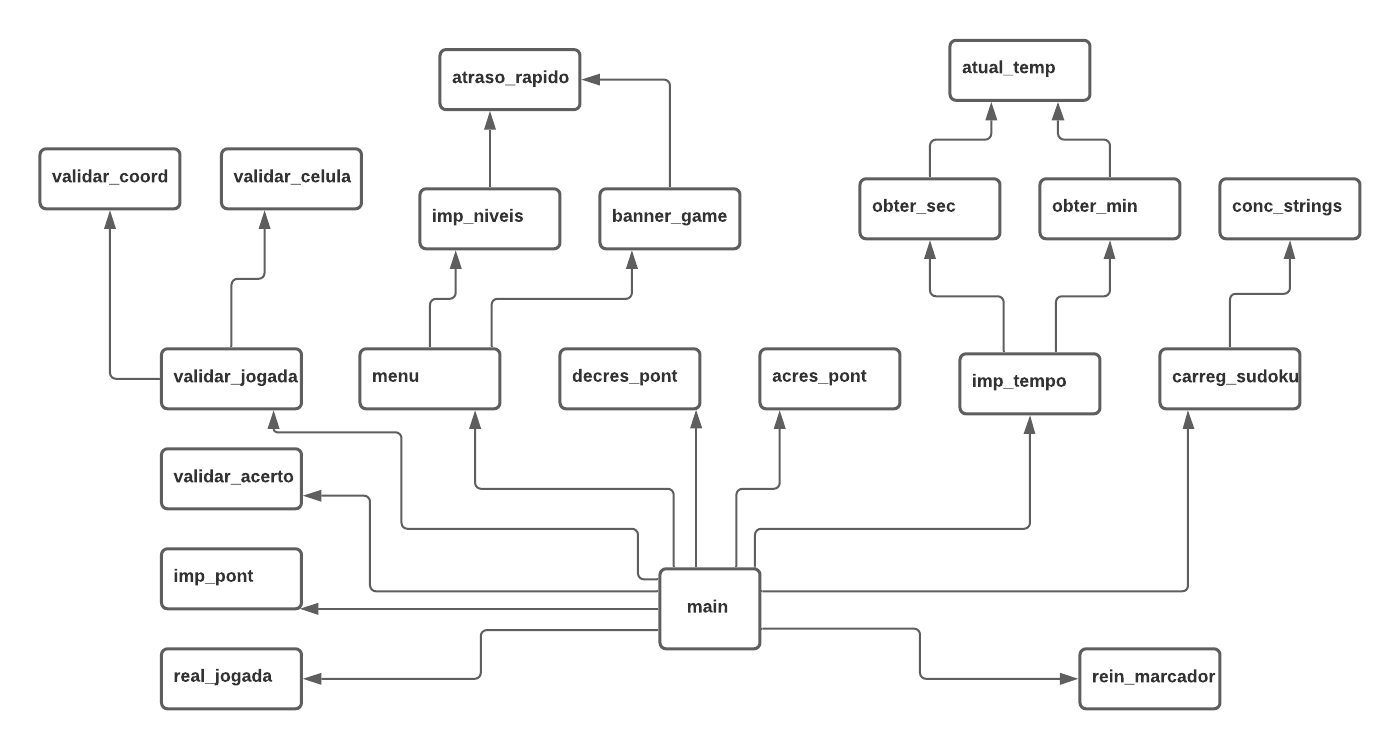


**3 - Algoritmo**

O algoritmo foi organizado a partir de diversas funções, cada uma com o seu propósito. As funções podem ser categorizadas de acordo com o espectro entre alto nível e baixo nível, pois algumas são meio-termo.

As de alto nível são aquelas mais importantes, que não dependem de nenhuma outra, a não ser que se trate de uma biblioteca. E as de baixo nível, ao contrário, são aquelas que dependem de outras funções para funcionar.

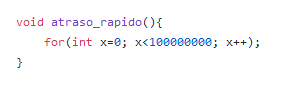
O diagrama abaixo ilustra a hierarquia de dependência entre as funções.



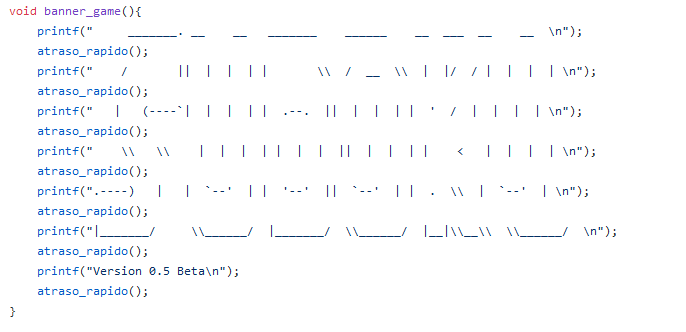
Observando o diagrama acima, podemos ver que a função *main* é a de mais baixo nível, dependendo de várias outras funções para executar da maneira desejada.

Ao total, foram criadas 25 funções, contando com a função principal. De forma resumida, e na ordem em que foram apresentadas no programa, são elas:

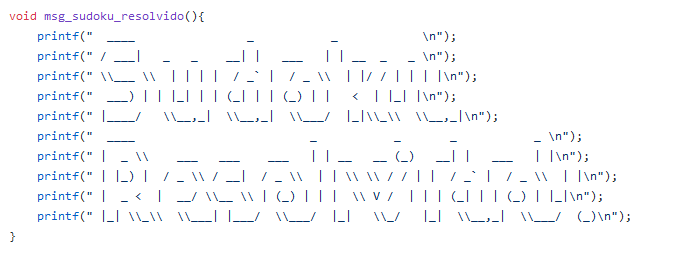
1. **Atraso\_rapido = usada para mostrar as mensagens do menu principal de forma pausada**



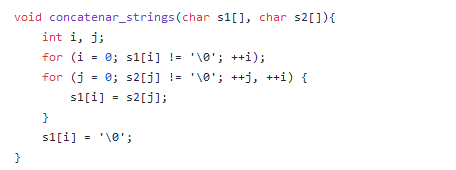
1. **Banner\_game = usada para imprimir na tela a mensagem principal no menu**



1. **Msg\_sudoku\_resolvido = mostra a mensagem estilizada do jogo resolvido**

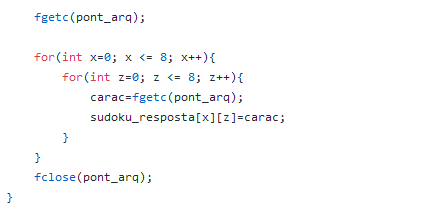


1. **Concatenar\_strings = concatena o caminho do arquivo com o nome do mesmo**

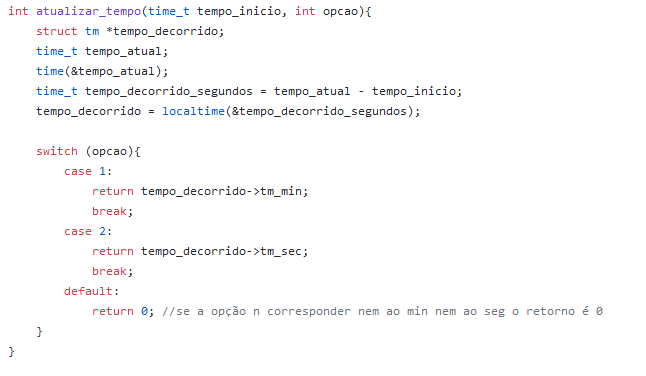


1. **Carregar\_sudoku =** Responsável por ler o arquivo onde o sudoku está armazenado e carregar o jogo

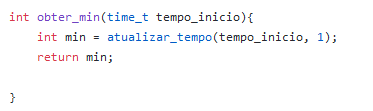




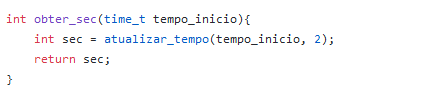
1. **Atualizar\_tempo =** Cronometro, retorna minuto se a opção for 0 e retorna segundo se for 1



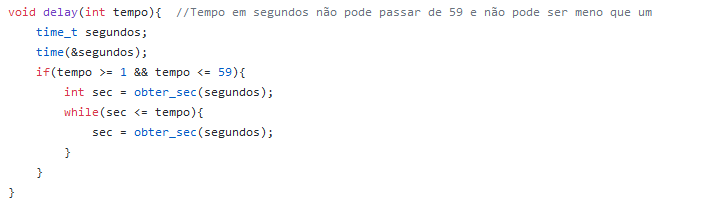
1. **Obter\_min = através de atualizar\_tempo, obtém o minuto atual.**



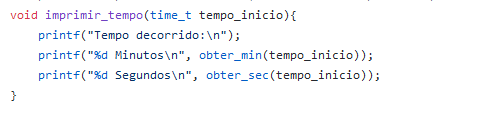
1. **Obter\_sec = através de atualizar\_tempo, obtém o segundo atual.**



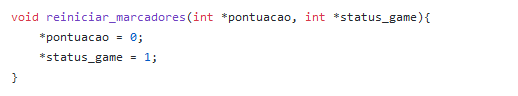
1. **Delay = Interrompe a execução do programa por uma fração de segundos.**



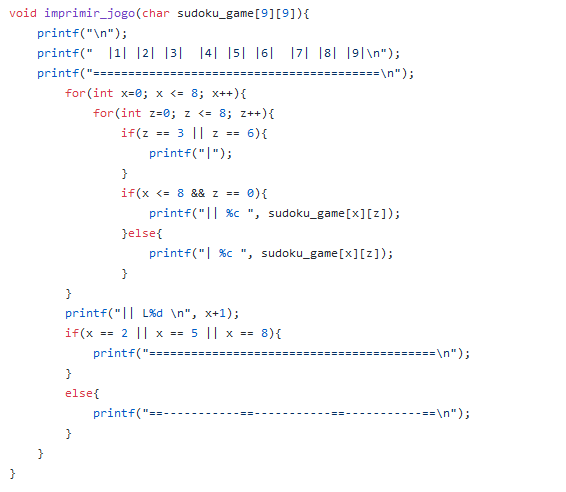
1. **Imprimir\_tempo = Captura o tempo decorrido da função atualizar tempo e o imprime.**



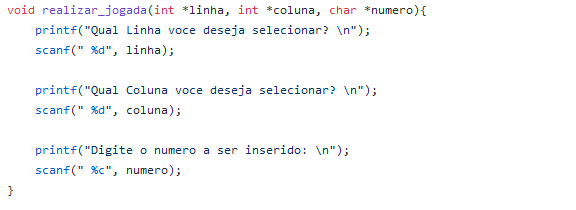
1. **Reiniciar\_marcadores = reinicia as variáveis com seus valores padrões para recomeçar o jogo.**



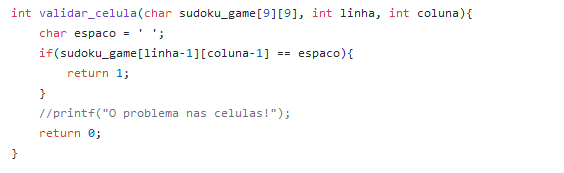
1. **Imprimir\_jogo = Percorre o vetor onde estão inseridos as jogadas do jogador, e as apresenta de maneira legível e amigável.**



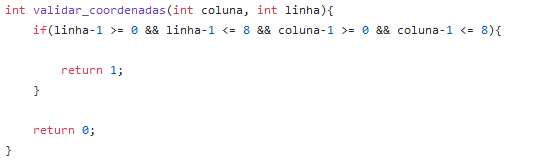
1. **Realizar\_jogada = Captura todas os dados da jogada.**



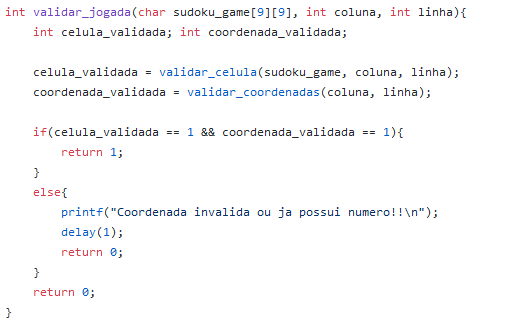
1. **Validar\_celula = Determina se a celula está disponível ou vaga.**



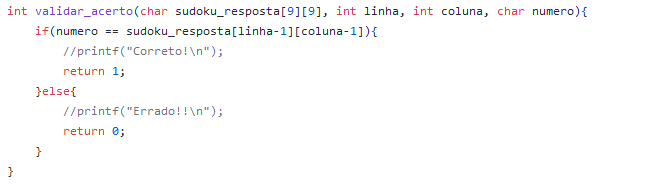
1. **Validar\_coordenadas = Determina se as coordenadas estão de acordo com os índices válidos do vetor.**



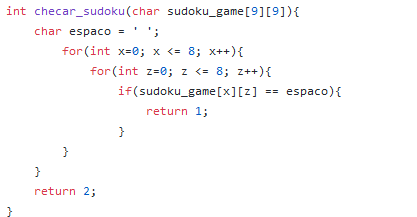
1. **Validar\_jogada = Reúne as principais funções de validação.**



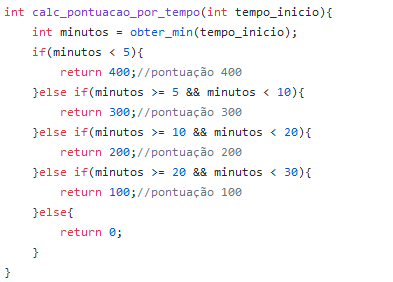
1. **Validar\_acerto = Determina se a jogada do jogador corresponde com o vetor que armazena o sudoku solucionado.**



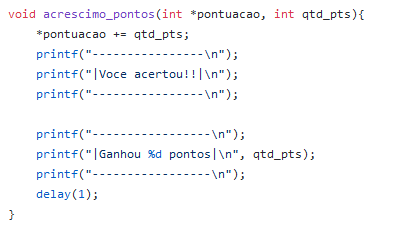
1. **Checar\_sudoku = A cada jogada checa se ainda possuem espaços vagos, se não tiver conclui-se que o jogador finalizou o sudoku, e o jogo termina.**



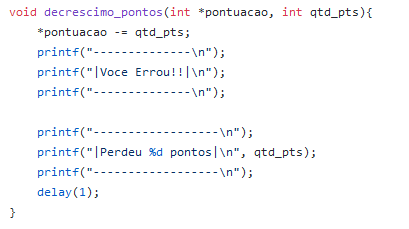
1. **Calc\_pontuacao\_por\_tempo = determina o bônus na pontuação em virtude do tempo decorrido até o término do jogo.**



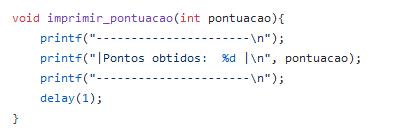
1. **Acrescimo\_pontos = exibe mensagem quando ganha pontos.**



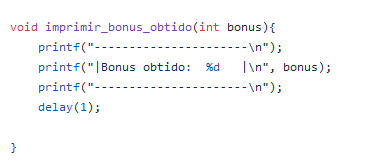
1. **Decrescimo\_pontos = Exibe mensagem quando perde pontos.**



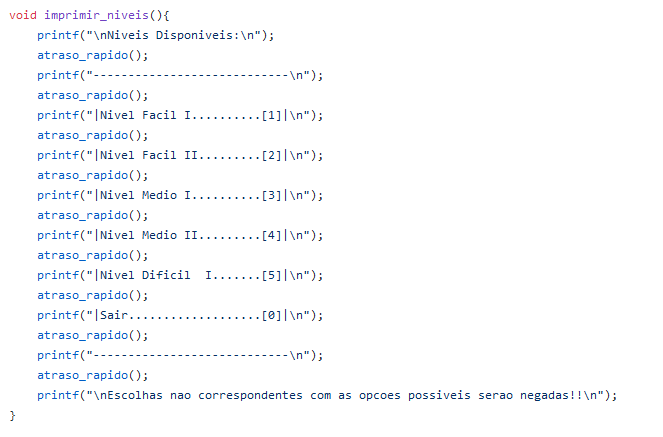
1. **Imprimir\_pontucao = mostra os pontos obtidos após a jogada.**



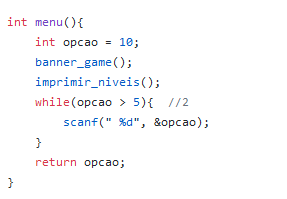
1. **Imprimir\_bonus\_obtido = mostra o bônus obtido no final ao completar o jogo.**



1. **Imprimir\_niveis = mostra o menu para escolha do nível do jogo.**



1. **Menu = opera o menu através da opção.**



1. **Main = função principal do jogo, responsável pela execução do mesmo, chamando as outras funções.**







**4 - Considerações Finais**

Durante o desenvolvimento do jogo aprendemos a organizar o código, através de funções específicas, que operam em conjunto para o funcionamento otimizado do programa, de forma a deixar o algoritmo refatorado e o mais legível possível.

Em consequência disso, podemos citar que justamente no desenvolver dessas funções, encontramos as maiores dificuldades, por serem muitas e interligadas.

Vimos na prática que a linguagem C é de médio nível, ao acessar recursos mais técnicos que outras linguagens não tem, como a manipulação de strings, que em C deve ser mais manual, pois deve-se determinar o fim dela, utilizando um caractere nulo.

O jogo é simples, porém alcançou o objetivo esperado, e funciona da devida forma.

**5 -** **Bibliografia**

https://blog.megajogos.com.br/sudoku-origem-historia-curiosidades/

Livro Arquitetura Limpa, página 75.