Jogo da Soma

Allen Hichard Marques dos Santos

Engenharia de computação – Universidade Estadual de Feira de Santana (UEFS)   
Caixa Postal 252 – 294 – Feira de Santana – BA – Brasil

Segunda solução de caso PBL

allenhichard@hotmail.com

1. **Introdução**

Emanuela era uma criança muito inteligente e vivia inventando novas brincadeiras na sua infância. Ela cresceu, casou-se e teve dois filhos, mas que ultimamente estão lhe dando muita preocupação, por não gostar de matemática e só passar o dia jogando os jogos eletrônicos que possuem.

Para tentar ajudar seus filhos, Emanuela teve uma maravilhosa ideia para unir o aprendizado e os jogos eletrônicos que seus filhos tanto amam. Ela pensou em um jogo chamado jogo da soma, um jogo de tabuleiro matemático, mas como não sabia programar, precisou de ajuda para desenvolver sua idéia e torná-la realidade.

O objetivo do problema é desenvolver um jogo de tabuleiro, onde pode ser jogado por dois jogadores. O jogo possui três níveis de dificuldades: fácil com dimensão 2x2; Médio com dimensão 4x4 e Difícil com dimensão 6x6, que é escolhido antes do início do jogo.

O jogo possui dois modos de jogar. Modo manual, onde o jogador informa um número e posiciona em uma das casas do tabuleiro. Modo aleatório, onde o computador sorteia o número e só pede ao jogador a posição do tabuleiro. São permitidos apenas números entre um e cem. Cada número só pode ser sorteado ou escolhido uma única vez.

No inicio do jogo, o tabuleiro começa com todas as casas zeradas e vai atualizando a cada jogada. O primeiro jogador joga com as linhas do tabuleiro e o segundo jogador joga com as colunas do tabuleiro. O primeiro jogador inicia o jogo. Alternam-se as jogadas até o tabuleiro ser completado. Calculam-se as somas de cada linha e de cada coluna do tabuleiro, se a maior soma for de uma das linhas o primeiro jogador é declarado vencedor, e se for à soma de umas das colunas o segundo jogador é declarado o vencedor. Em caso de empate, verifica a segunda maior soma e assim por diante, após conferir todas as somas e não houver um vencedor é declarado empate.

1. **Metodologia**

O software foi desenvolvido em linguagem de programação “C” com algumas restrições. Desde a primeira sessão foram levantadas questões sobre modularização e como retornar parâmetro de funções, deixando claro que só poderia usar variáveis locais nesse problema.

O uso de variável global não foi permitido. Porque ela permite que todas as funções acessem suas informações, tornando não seguro para o código, por isso os parâmetros para passar informações precisas com maior segurança.

As funções para torna o software bem modularizado, ficou determinado como globais, pois, outra função diferente da função principal pode precisar de informações de outra função. Um software bem modularizado é um software dividido em várias funções. Essas funções devem ser globais ou locais, dependendo do objetivo do programador.

Após as restrições foram passados estudos sobre como modularizar um código; usar apenas uma parte da matriz; somar linhas e colunas da matriz com vetores; diferença entre função e procedimentos; organização das somas das linhas e colunas.

Nas demais sessões foram para elaborar o algoritmo do programa, para enxergar dúvidas que ainda não foram levantadas nas reuniões. Foram encontradas dúvidas sobre: envio de parâmetro; diferença de função void para inteira; ordenar as somas dos vetores em ordem crescente ou decrescente.

A passagem de parâmetro deixou clara a diferença de função void e inteira, já que a void não retorna valor, quanto à função inteira devolve um único valor inteiro, valor que pode ser enxergado por outra função. Matriz e vetores não precisam ser retornados, pois se tratam de ponteiros que acessam através do endereço de memória uma variável.

Na ordenação foram pesquisados três tipos: BubbleSort; InsertionSort; SelectionSort, códigos que já possuem características de ordenação, onde não ficou restrito o uso de nenhum deles, então cada programador pode escolher o melhor por afinidade que achar e utilizá-lo em seu software.

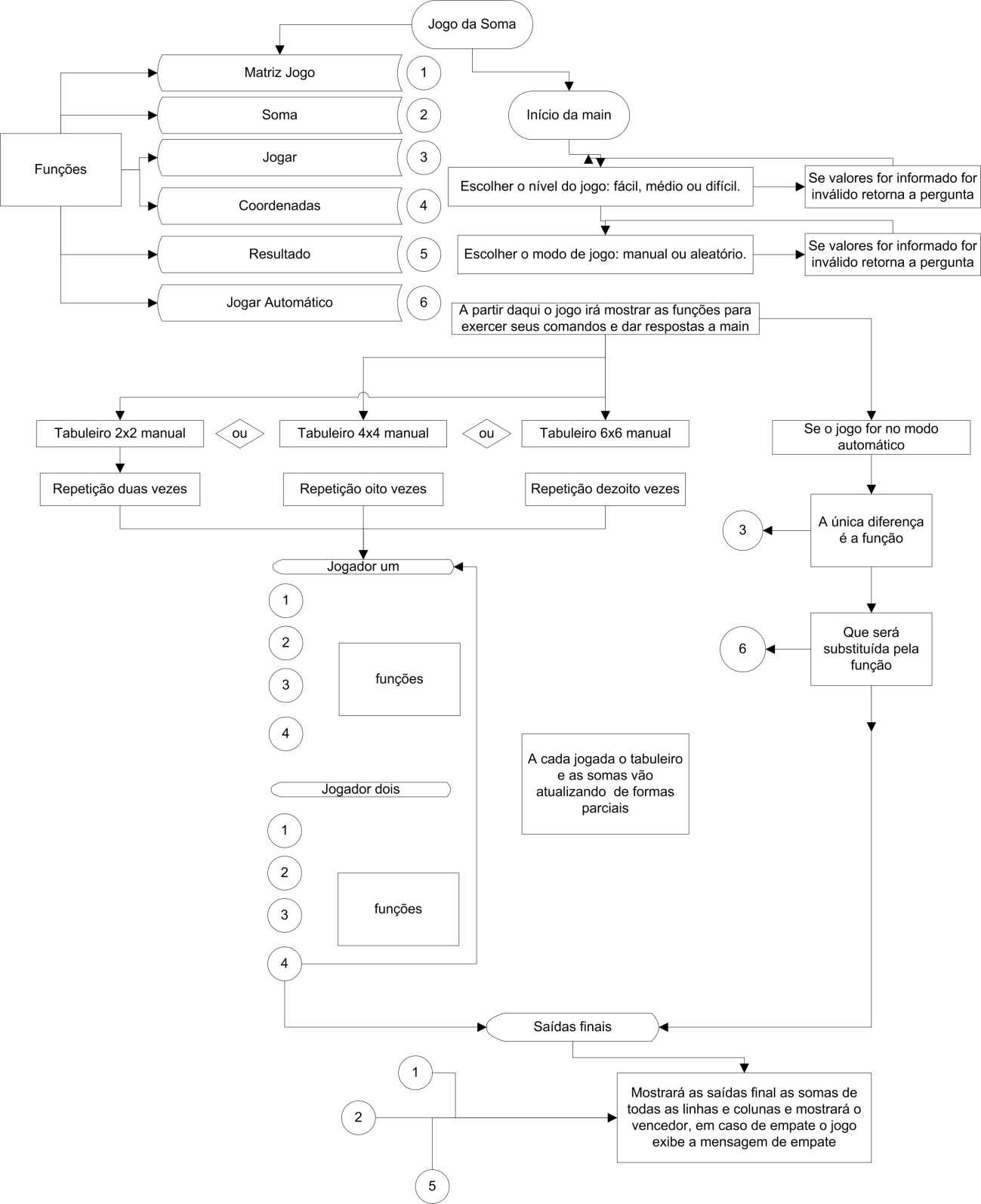
**Requisitos/funcionalidades para o software**

O software tem funcionalidade de permite que o usuário escolha o nível de jogo, que pode ser: fácil, médio ou difícil e seu modo de jogar: manual ou aleatório. Mostrar o tabuleiro a cada jogada atualizado e suas somas parciais. Só receber números entre um e cem, não podendo repetir nenhum número já jogado. O tabuleiro só aceita o número se for uma coordenada existente, e que nesta posição não contenha um número já indicado anteriormente.

O tabuleiro é jogado por dois jogadores que iram alternando suas jogadas até completá-lo. Finalizando o jogo, mostra o tabuleiro completo, as somas finais de cada linha e coluna do tabuleiro, mostrando o vencedor com a maior pontuação de linhas ou colunas.

O jogo tem uma funcionalidade especial para empates, caso a primeira maior soma for igual entre linhas e colunas, é considerada a segunda maior até testar todas as linhas e colunas. Caso o jogo não tenha um vencedor é considerado um empate, finalizando o jogo.

**Na figura 1.** Mostra a ordem de codificação em forma de fluxograma, para entender melhor como foi construído o software do jogo da soma.



**Figura 1. Fluxograma do Algoritmo.**

**Ordem de codificação**

A construção do algoritmo foi feita de forma bem modularizado, fazendo repetições de códigos tornarem desnecessárias. Foram feitas funções para ler e atualizar o tabuleiro; somar suas linhas e colunas, jogar no modo manual ou aleatório, pedir suas coordenadas e mostrar o ganhador do jogo.

Foram declaradas seis funções, com finalidade de receber e devolver informações para a main, todas tem protótipos para ter maior controle de informações e maior agilidade na programação, além de especificar cada uma para um determinado comando. Como foi programado primeiro a main, tornou necessário saber quantas funções o software precisaria por isso os protótipos.

Ao iniciar o jogo, pede aos jogadores que escolham o nível de jogo e o modo de jogar. No nível do jogo pode escolher três opções para jogá-lo. No modo de jogo pode escolher duas opções para jogá-lo.

Independente do nível e modo de jogo as condições a serem executadas são as mesmas, com uma pequena alteração, dependendo do nível o tabuleiro é aumentado e de acordo com o modo de jogo, altera-se as funções jogar e jogarautomático.

Na inicialização da função principal foi escolhido uma matriz [6] [6] e vetores para linhas [6] e coluna [6], para que se possa manipulá-los. O usuário no Menu escolhe o nível de jogo, onde um número é enviado como parâmetro para as funções, que pode ser de 2 para nível fácil, 4 para nível médio e 6 para o nível difícil.

As matrizes e vetores foram inicializados com valor máximo pedido no problema, para poder manipular e usar apenas uma parte dela. Na construção do código foi feito todo o código sem modularização 2x2. Como os demais níveis são repetição de códigos só mudando o valor da matriz, ficou mais fácil manipulá-la com uma variável o seu tamanho.

Iniciando o jogo, a matriz é mostrada zerada, para que o usuário informe as coordenadas para jogar o número que pode ser aleatório ou manual. Após o usuário informar as coordenadas, o número é atualizado na matriz, fazendo suas somas parciais.

No final, o tabuleiro é completado, mostrando as somas finalizadas e revelando quem foi o vencedor. Em caso de empate, tem que ser conferidos as somas de todas as linhas e colunas, não existindo um ganhador é considerado empate. Para facilitar as conferencias das somas de linhas e colunas foram organizadas de forma decrescente pelo algoritmo: SelectionSort, e mostrado aos jogadores suas somas de formas organizadas, feitas para facilitar a verificação dos jogadores.

Após modularizar o código da matriz [2] [2], para todas as matrizes com ajuda de um parâmetro, surgiu dois problemas:

O primeiro foi quando o usuário informava um número existente na matriz ele só conferia uma vez, porque entrava em uma repetição pedindo um novo número, mas não verificava se era entre 1 e 100 e se existia na matriz, como essa mesma função era responsável de fazer essa atividade, usei a recursividade, para caso o número existir na matriz, refazer a mesma função;

O segundo foi nas coordenadas, o código analisava se a coordenada já tinha um número informado anteriormente, caso positivo ela pedia uma nova coordenada, podendo colocar em uma coordenada não existente, então foi colocado duas repetições para linhas e colunas estarem sempre analisando se a posição existe.

O software foi desenvolvido em um sistema operacional Windows, foi utilizado o DEV C++, que é um compilador de códigos de linguagem de programação “c” e “c++”. O fluxograma foi feito utilizando o software da Microsoft Visio, que é um programa que constrói fluxogramas, além disso, para ter certeza do completo funcionamento, foi testado o código no sistema operacional Linux, como tem algumas condições que não existem no Linux a exemplo de algumas bibliotecas, foram feitos apenas adaptações para o teste, mas não houve modificações de condições do código.

1. **Resultados e discussões**

O programa só é garantindo seu completo funcionamento em plataforma Windows. Na execução do software, existem opções de jogo que o menu trás para o usuário indicar seu nível e modo de jogo. O nível de jogo tem três escolhas possíveis de jogo: 1 para fácil; 2 para médio e 3 para difícil. E no modo de jogo tem duas opções: 1 para manual e 2 para automático. Caso o usuário informe algum número não correspondente, as perguntas serão refeitas.

Para que o programa seja executado sem nenhuma interrupção precisa de algumas entradas válidas, para não fazer o usuário ter que está repetindo informações, no menu do programa as entradas válidas são 1, 2 ou 3 para os níveis de jogo, e 1 ou 2 para seu modo de jogo. No tabuleiro, números entre 1 e 100, não podendo repetir números já utilizados. As coordenadas tem que existir na matriz, que varia de acordo com seu nível, o zero é considerado uma coordenada válida, e não pode indicar uma coordenada já jogada.

**Saídas do programa**

O programa a cada jogada tem saídas parciais, para auxiliar os jogadores a raciocinar seu modo de jogar que seria:

* Qual o jogador atual;
* O estado atual do tabuleiro;
* Somas parciais de todas as linhas e colunas.

No final do jogo as saídas são:

* O tabuleiro totalmente preenchido;
* As somas das linhas e colunas;
* Todas as somas das linhas e colunas organizadas de forma decrescente;
* O jogador vitorioso;
* A soma alcançada pelo jogador vitorioso.

**Testes efetuados e resultados obtidos**

No modo manual, independente do nível de jogo, foram feitos testes para verificar o funcionamento do programa, testes aos quais garantiram aos jogadores um jogo eficaz.

Primeiro teste: jogar números menores que 1 e maiores que 100, o jogo em todas as tentativas não permite a entrada deste número;

Segundo teste: jogar um número já existente no tabuleiro, foi preenchido metade do tabuleiro, depois foi jogado todos os números que tinha nesse tabuleiro, e nenhum dos números conseguiram se posicionar no tabuleiro;

Terceiro teste: as coordenadas só permitem posições existentes no tabuleiro, foi tentando indicar uma coordenada não existente, mas não foi permitida a entrada deste número;

Quarto teste: colocar um número em uma coordenada existente, mas que já contenha número irá pedir novamente as coordenadas.

**Em que situação o seu programa não funcionaria**

Todos os requisitos estão funcionando por completo, mas só tem uma entrada especifica que apresenta erros inesperados. Foram feitos testes exclusivos em caso de digitando uma letra ao invés de números e em qualquer situação que for digitado uma letra, o funcionamento do programa dará respostas inesperadas, ocasionando em erro de execução, ”em hipótese alguma pode ser utilizado letras”.

1. **Conclusão**

Após todos os testes realizados, todo o funcionamento que o problema exigiu foi cumprido, números e coordenadas válidas e as saídas esperadas para o usuário, além disso, foram feitos algumas condições adicionais, para que o programa funcione de forma correta, a exemplo do menu, que fica em repetição até informado um número de entrada válido.

Foi colocada na interface a indicação das linhas e colunas, para facilitar o entendimento do usuário e mostrar a saída das somas das linhas e colunas do tabuleiro organizadas decrescente, para o usuário ter a facilidade de conferir as somas para comparar linha e coluna e ver se o resultado do ganhador é válido, também não deixou de ser feito nada explicito no problema, todos os requisitos foram atendidos e incrementado.

As melhorias propostas seria salvar um histórico de ganhadores. Ter um percentual de partidas jogadas em cada nível e em cada modo de jogo, fazendo uma continuidade de jogadas tornando o jogo mais emocionante e competitivo. Opção de zerar contador de jogos já realizado ou adicionar caso um novo oponente. Opção para iniciar um novo jogo sem ter a necessidade de executar novamente a cada jogada.

1. **Referência**

Schildt, Herbert “C: completo e total, Terceira edição”.

Lopes, Anita e Guto Garcia “introdução à programação: 500 Algoritmos resolvidos, 5ª

“tiragem”.