Relatório de implementação do jogo "2048"

Cauê Scotti, Lucas Campos Machado

Resumo. *O presente relatório tem o objetivo de explicar e detalhar o funcionamento e implementação do jogo "2048", suas funções e estruturas, como atividade final na cadeira de algoritmos e programação.*

Abstract. *The present report aims to explain and detail the operation and implementation of the game "2048", its functions and structs, as final activity in the couse of algorithms and programming*

Variáveis principais

A representação da matriz principal do jogo é feita através de uma struct, tendo seu tipo definido pelo nome "Matriz", que possui dois ponteiros: o primeiro aponta para o valor de uma determinada casa e o segundo para sua cor. Na implementação, foi definido uma matriz do tipo "Matriz", chamada durante a execução de "var", com quatro linhas e quatro colunas, representando assim todas as dezesseis casas que o jogo "2048" possui.

Como a variável de representação principal das casas é definida por uma matriz de ponteiros, foram adicionados também vetores que contenham os possíveis valores que serão apontados por esses ponteiros. O vetor "val" armazena todos os valores das potencias de dois, iniciando em dois em terminando em dois mil e quarenta e oito, com um ponteiro "pont" que aponta para seu início. O vetor "cores" possui doze valores inteiros, que serão interpretados como diferentes cores pelas funções da biblioteca conio (valores definidos arbitrariamente pelos programadores).

Para auxiliar no movimento dos componentes da matriz, foi criado também uma matriz auxiliar, com quatro linhas e oito colunas. A função dessa matriz é conter os valores inteiros de cada casa entre a coluna um e quatro e, entre as colunas cinco e oito, conter os valores da execução anterior; essa funcionalidade será útil para verificar se houve mudança nas casas de uma execução para outra e também para desfazer jogadas.

Por fim, a variável "pontuacao" guarda a pontuação atual do jogo e a variável "movimento" a quantidade de movimentos realizados até então. Além disso, outras variáveis foram definidas para controlar as repetições do jogo e a execução ou não de certas funções. São elas: "i", "j", "jogo", "opcao", "teste", "comando", "novo_jogo", "salva_jogo", "compara" e "testa_pontuacao".

1. Structs

1.1 typedef struct tiles { } Matriz;

A presente estrutura é utilizada na representação dos valores de cada casa, bem como na cor que cada uma possui quando for impressa na tela. Para tal, duas variáveis a compõem: "int *p_casa" e "int *p_cor".

O ponteiro "p_casa" aponta para o módulo de determinada casa. Os valores possíveis são encontrados no vetor "val", que vai de dois a dois mil e quarenta e oito. Caso a casa analisada tenha valor zero, o ponteiro "p_casa" aponta sempre para NULL.

O ponteiro "p_cor" aponta para a cor que a casa possui, contidas no vetor "cores". Diferente do ponteiro anterior, se o módulo da casa for zero, "p_cor" não apontará para NULL, mas sim para um valor de "cores", uma vez que a casa nula na exibição do jogo possui uma cor também.

2. Funções

2.1 Funções de movimento

Este grupo de funções abrange todos os comandos necessários para que os movimentos sejam executados de maneira eficaz tal como segue:

void Inclui_Elemento

No início do jogo, tanto a matriz de ponteiros da struct como a matriz auxiliar são inicializadas com valores NULL e zero, respectivamente. A função "Inclui_Elemento" recebe como parâmetro a matriz auxiliar e o ponteiro que aponta para a mesma e adiciona aleatoriamente um valor entre dois e quatro (10% de chance de ser quatro) em uma casa também aleatória.

A função sorteia um valor entre dois e quatro e em seguida um valor para a coluna e outro para a linha. Se a casa da linha e coluna sorteadas esteja vazia, o valor sorteado anteriormente é anexado. Caso contrário, o processo se repete, até que seja sorteada uma casa vazia.

int Movimento_Matriz

A função "Movimento_Matriz" é responsável por executar o movimento das casas para as quatro posições possíveis e por calcular o a pontuação após cada execução. Recebe a matriz auxiliar, seu ponteiro e o movimento informado.

Para que o movimento seja realizado, a função analisa linha por linha ou coluna por coluna, dependendo do movimento. O primeiro vetor "linha" recebe os valores da linha ou coluna alocados no canto desejado (índices maiores para a direita ou baixo e menores para a esquerda ou cima) e adiciona zeros para completar o tamanho correto. Por exemplo, se a linha é {2, 0, 2, 0} e o movimento solicitado é para a direita, o vetor "linha" fica na forma {0, 0, 2, 2}.

Após, a função verifica se existem números iguais um ao lado do outro. Se sim, soma-os e realiza o procedimento novamente, para que a linha fique sem espaços vagos. Utilizando o mesmo exemplo acima ({0, 0, 2, 2}), é possível perceber que o número dois está ao lado de outro dois. Sendo assim, ambos serão somados, jogados para a direita e

por fim alocados na matriz original. O ponteiro da matriz é utilizado para que os valores sejam corretamente passados para a matriz.

Caso ocorra alguma soma de valores, essa soma é adicionada à variável "pontos" e retornada no fim da função.

void Atualiza_aux

"Atualiza_aux" é a função que atualiza os dados da matriz de ponteiros do tipo struct (que contém os valores das peças) bem como da matriz auxiliar (utilizada em "Movimento_Matriz"). Recebe o ponteiro da matriz auxiliar, a matriz auxiliar, a matriz de ponteiros do tipo struct, o ponteiro dos possíveis valores (descrito no trecho 'Variáveis Principais'), o vetor de cores e um inteiro entre um e três, chamado de "menu".

Essa função pode ser utilizada de três formas diferentes, dependendo do valor que é passado na variável "menu". Se menu é igual a um, os valores contidos na matriz auxiliar são passados para a matriz de ponteiros, que aponta para NULL caso o valor seja zero ou aponta para os outros possíveis valores das potências de dois, atualizando também as cores que serão impressas na tela junto com os valores de cada casa.

Se menu é igual a dois, a função passa os valores contidos entre as colunas um e quatro da matriz auxiliar para as colunas cinco a oito. Assim, é possível saber a disposição dos valores das casas uma execução atrás. Essa funcionalidade é vital para conferir se uma jogada é legal e para desfazer jogadas.

Se menu é igual a três, a função passa os valores das colunas cinco a oito para as colunas um a quatro e atualiza também a matriz de ponteiros. Esse processo permite que as jogadas sejam desfeitas, já que os valores das colunas cinco a oito da matriz auxiliar contém os valores da execução anterior.

int Compara_Anterior

Essa função serve para comparar a matriz anterior (localizada entre as colunas cinco e oito da matriz auxiliar) com a matriz atual após a execução do movimento (colunas um a quatro). Caso os valores sejam diferentes, a função retorna um, indicando que o movimento é valido. Se os valores forem iguais, significa que não é permitido executar o movimento na direção selecionada, retornando zero.

int Confere_Matriz

A função "Confere_Matriz" é a função responsável por verficar se o jogo acabou. A função recebe a matriz de ponteiros e a verifica elemento por elemento.

São dois critérios levados em consideração para saber se o jogo já terminou. O primeiro é se o valor 2048 for alcançado e o segundo é se não existirem mais espaços em branco no tabuleiro. Caso alguma dessas condições for atendida, retorna o valor zero, indicando que o jogo acabou. Caso contrário, retorna o valor um.

2.2 Funções de arquivos

Este grupo de funções tem dois propósitos: salvar as dez maiores pontuações alcançadas e salvar e carregar o tabuleiro quando solicitado.

int Confere_Pontuacao

A presente função tem por objetivo comparar a pontuação atingida quando o jogo acabar com as pontuações registradas em um arquivo texto. Sendo assim, a função recebe a pontuação no fim do jogo.

As dez melhores pontuações atingidas são salvas em um arquivo do tipo texto chamado "pontuacoes.txt", onde cada linha contém o nome digitado pelo usuário que atingiu a pontuação com 20 caracteres, um caractere do tipo ' | ' e a pontuação em seguida, como na imagem abaixo:

```
Teste 1             |66
Teste 2             |60
Teste 3             |58
Teste 4             |56
Teste 5             |54
Teste 6             |52
Teste 7             |50
Teste 8             |49
Teste 9             |46
Teste 10            |44
```

Com as pontuações salvas dessa forma, uma string "linha" é utilizada para ler cada uma das linhas do arquivo e, utilizando a tokenização para obter a pontuação, a função compara cada resultado com os pontos alcançados no jogo. Se a pontuação do último alcançada for maior que alguma pontuação salva ou se o arquivo não contém dez linhas (logo, a pontuação será salva de qualquer maneira) ou se o arquivo ainda não existe, "Confere_Pontuacao" retorna o valor um, informando que a pontuação atual obtida deve ser salva.

void Salva_Pontuacao

Se for constatado que a pontuação obtida no fim do jogo é maior que alguma pontuação salva, a função "Salva_Pontuacao" recebe essa pontuação e a salva no arquivo "pontuacoes.txt" em ordem decrescente.

Primeiramente, a função solicita ao usuário que digite um nome para salvar. Em seguida, esse nome é padronizado na string "nome_real", para que o arquivo mantenha sempre a mesma quantidade de caracteres em cada linha. Em seguida, o arquivo "pontuacoes.txt" é aberto para leitura e é criado um arquivo auxiliar "pontuacoes_aux.txt". Todas as linhas de "pontuacoes.txt" têm sua pontuação conferida como na função "Confere_Pontuacao" e são salvas no arquivo auxiliar até que a pontuação adquirida no jogo atual seja maior que as salvas. Nesse momento, a função

salva o nome e a pontuação atual e termina de salvar as outras linhas até que o arquivo auxiliar tenha dez linhas. Por fim, todas as linhas do arquivo auxiliar são passadas para o arquivo original. Esse processo é necessário para alocar a pontuação atual na posição certa e deixar o arquivo original sempre com dez linhas.

void Salva_Tabuleiro

A função "Salva_Tabuleiro" é utilizada quando o usuário deseja gravar o jogo atual. Para tal, a função recebe a matriz de ponteiros, que contém o valor de cada casa, a pontuação atual e a quantidade de movimentos.

Esses valores são salvos em arquivos binários. Sendo assim, os valores da matriz atual são todos passados para um vetor de tamanho dezesseis. Em seguida, um arquivo de nome "ultimo_tabuleiro.bin" é criado. Então, os sessenta e quatro bytes do vetor (quatro bytes para inteiros vezes o tamanho dezesseis) são adicionados ao arquivo, logo após os quatro bytes da pontuação e por fim os quatro bytes da quantidade de movimentos.

void Carrega_Tabuleiro

"Carrega_Tabuleiro" é responsável por carregar um jogo salvo anteriormente. Para que essa tarefa seja executada, a função recebe apenas ponteiros, que apontam para a matriz auxiliar, para a variável que guarda a pontuação e para a variável que guarda a quantidade de movimentos.

A partir de um arquivo com um jogo salvo, a função lê todos os valores e os transfere para as variáveis atualmente utilizadas. Como citado na função "Salva_Tabuleiro", o arquivo binário que guarda o jogo tem seus primeiros sessenta e quatro bytes com os valores da matriz. Sendo assim, o arquivo é lido de quatro em quatro bytes, já que o tipo inteiro possui quatro bytes, e os valores lidos são anexados à matriz auxiliar, percorrendo todas as linhas e colunas. Após as dezesseis leituras de quatro bytes, os últimos oito bytes possuem a pontuação e os movimentos salvos, nessa ordem. Portanto, são realizadas mais duas leituras de quatro bytes e os valores são anexados à variável da pontuação e à variável da quantidade de movimentos.

2.3 Funções de impressão

As funções de impressão são as que definem a maneira como o jogo é apresentado na tela, sendo algumas também solicitam e retornam as escolhas do usuário no decorrer das execuções do programa. Para esse propósito foi utilizada a biblioteca conio.h e a representação é feita em modo texto.

int Menu_Inicial

Esta função imprime a tela de início do jogo, aguarda que uma das três opções iniciais sejam selecionadas e a retorna depois de informada.

Os itens que aparecem na tela são uma escrita de "2048", que inicia na coluna vinte e seis linha dois e termina na coluna oitenta e sete linha oito, e uma caixa, que inicia na

coluna vinte e seis linha quatorze e termina na coluna oitenta e nove linha vinte e cinco, que contém as opções de início de jogo do usuário.

void Print_Fundo

A função "Print_Fundo" muda a cor de fundo da tela de comandos, alterando da cor preta padrão para a cor branca.

void Print_Base

A função "Print_Base" imprime a base do jogo, ou seja, o tabuleiro, que inicia na coluna oito linha quatro e encerra na coluna cinquenta e seis linha vinte e oito, as escritas "pontuacao: " e "movimentos: " no canto superior direito, a escrita "Melhores pontuacoes: " no centro a direita e os possíveis comandos no canto inferior direito.

void Print_Pontuacoes

Esta função é utilizada para imprimir no lado direito da tela, abaixo da escrita "Melhores pontuacoes: ", as três melhores pontuações salvas no arquivo "pontuacoes.txt".

A impressão é feita exatamente como está no arquivo, caractere por caractere sem nenhum tipo de formatação. Por esse motivo, todas as formatações necessárias já são realizadas no momento de salvar a pontuação na função "Salva_Pontuacao".

void Print_Matriz

"Print_Matriz" é a função que faz a leitura dos valores da matriz principal e das cores atribuídas a cada casa e os imprime dentro do tabuleiro de jogo. A função recebe a matriz de ponteiros do tipo struct, que contém os valores de cada casa, bem como suas respectivas cores, a pontuação atual e a quantidade de movimentos.

A função lê cada valor da matriz principal e a cor que cada um possui. Cada casa do tabuleiro possui um tamanho de onze colunas e cinco linhas. Se o número for diferente de zero, "Print_Matriz" lê a cor da casa, aplica na função "textbackground()" da biblioteca conio.h e imprime um retângulo da cor lida com o valor da casa no centro do retângulo. Se o número for zero, então "Print_Matriz" lê a cor da casa, aplica na função "textbackground()", imprime o retângulo, mas não imprime zero no centro, uma vez que a casa está vazia.

Por fim, a função ainda imprime os valores recebidos da pontuação e da quantidade de movimentos ao lado das escritas "Pontuacao: " e "Movimento: " no lado direito, respectivamente.

int Print_Caixa

Nos momentos em que alguma escolha do usuário deve ser confirmada ou quando o jogo acaba, a função "Print_Caixa" imprime na tela uma caixa que solicita que o usuário confirme a decisão informa que o jogo foi encerrado.

O tipo de informação que aparecerá na caixa é definido pelo parâmetro que é passado para a função, que recebe uma variável "menu". Se menu é igual a zero, a função

solicita que o usuário confirme se quer começar um novo jogo. Se menu é igual a um, a função solicita que o usuário confirme se quer salvar o jogo atual; se sim, informa que o jogo foi salvo. Se menu for diferente de qualquer um desses valores, a função imprime que o jogo foi encerrado.

A função também retorna o valor quarenta e nove (dígito um em ASCII) caso a escolha seja positiva e um valor diferente se caso a escolha tenha sido negativa.

3. Função Main

Além de conter todas as variáveis citadas em "Variáveis Principais" nesse documento, a função main organiza a execução de todas as funções apresentadas através de comandos de repetição e condicionais.

3.1 while (jogo)

Este loop é o responsável por manter o jogo executando até o momento em que o usuário deseje sair do jogo. A variável "jogo" é inicializada com o valor um e recebe o valor zero caso, na tela inicial, seja selecionada a opção "Sair".

3.2 if (opcao == 49 || opcao == 50)

A variável "opcao" recebe o valor refente à tabela ASCII do caractere digitado na tela inicial, sendo quarenta e nove para um e cinquenta para dois. Essa transformação acontece devido ao uso da função "getch", que armazena caracteres ou o valor ASCII dos mesmos, podendo ser encontrada durante todo o desenvolvimento do código.

Se "opcao" for igual a quarenta e nove, um novo jogo é iniciado, iniciando o tabuleiro apenas com valores nulos mais dois aleatórios para o começo do jogo, passando para o próximo comando de repetição. Se "opcao" for igual a cinquenta, o código carregará o jogo salvo em arquivo binário e seguirá para o próximo comando de repetição normalmente.

3.3 do { } while (teste)

A variável "teste" é a variável que define se o jogo ainda pode ser continuado através da verificação das condições pela função "Confere_Matriz". Após cada execução, a função confere a matriz e retorna o resultado para "teste", sendo zero quando não for mais possível continuar o jogo.

O primeiro passo dessa repetição é armazenar o comando desejado pelo usuário, atribuindo o valor à variável "comando". Em seguida, uma série de condicionais são verificados com a variável "comando". São elas:

* if (comando == 110) ('N' em ASCII)

Se for digitado o caractere 'N' no início da repetição, o programa solicita uma confirmação ao usuário e, se positiva, apaga o jogo atual e começa um novo jogo.

* else if (comando == 115) ('S' em ASCII)

Se for digitado o caractere 'S' no início da repetição, o programa também solicita uma confirmação ao usuário e salva o jogo atual em um arquivo binário caso o comando seja confirmado.

* else if (comando == 27) ('ESC' em ASCII)

Se for digitado o caractere 'ESC', o programa informa que o jogo é fechado e retorna para a tela inicial. Para que isso aconteça, o loop de "teste" deve ser finalizado. Sendo assim, caso esse comando seja digitado, "teste" recebe o valor zero, encerrando o loop.

* else if (comando == 8) ('BACKSPACE' em ASCII)

A tecla 'BACKSPACE' é responsável por desfazer uma jogada. Ou seja, se for informada no início da repetição, o tabuleiro, a pontuação e a quantidade de movimentos voltam para a posição anterior.

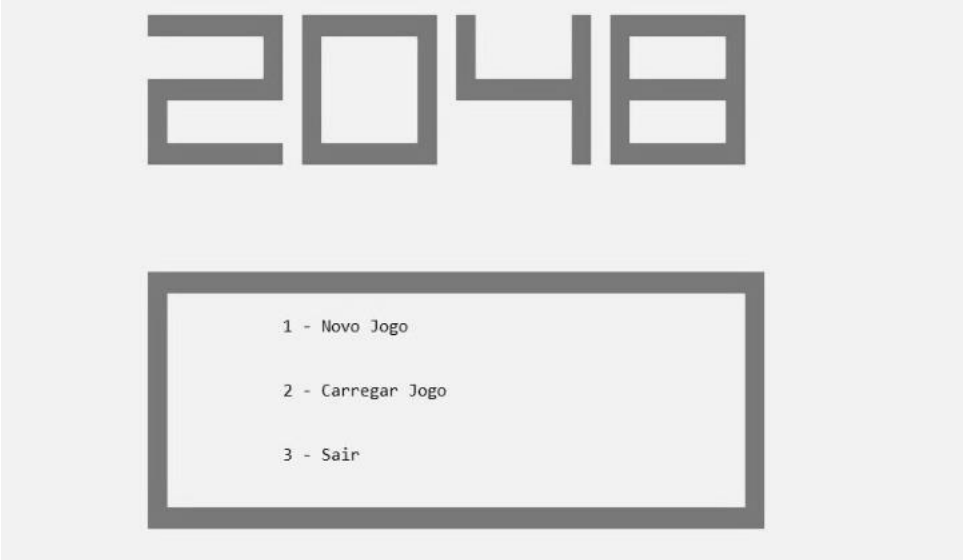
*else (Direcionais)

Por fim, se algum direcional for digitado, o programa realiza o movimento do tabuleiro no sentido solicitado e verifica se houve uma mudança no estado anterior da matriz para o estado atual. Se sim, o movimento é executado, a pontuação atualizada e um novo elemento incluído; se não, o tabuleiro permanece como está. Por fim, é realizada a verificação da condição de jogo e, se for constado que não é possível continuar o jogo, a mensagem de encerramento aparece e, caso a pontuação seja maior que as dez maiores, é salva no arquivo texto.

4. Instruções de uso

4.1 Tela inicial

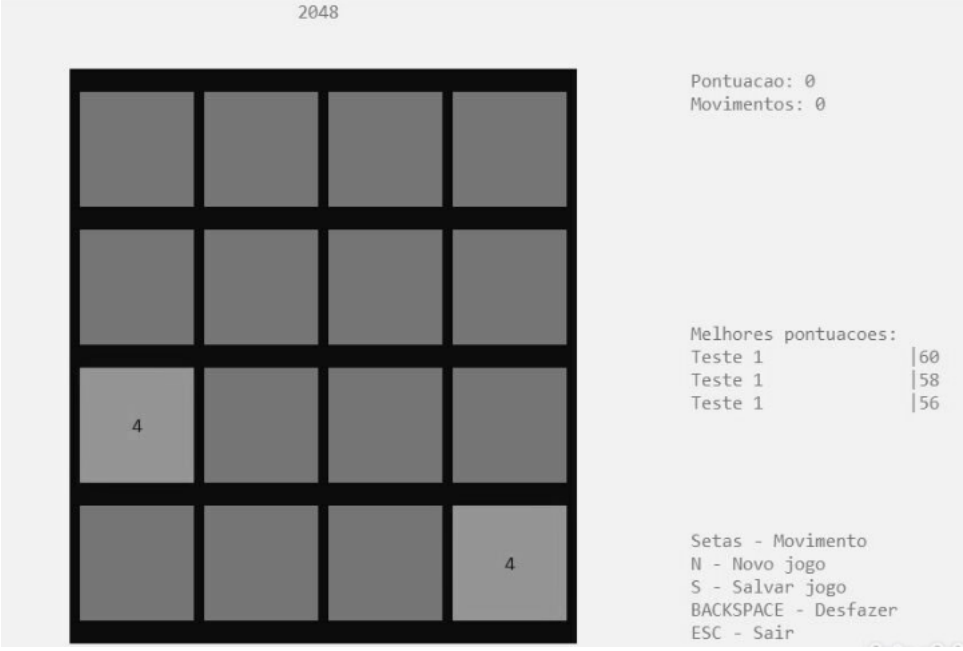
A tela inicial é onde a execução do programa começa. Neste primeiro momento são mostradas as primeiras opções ao usuário como segue:



Para selecionar alguma opção, basta digitar o número referente à opção desejada. "Novo jogo" inicia um novo jogo, "Carregar jogo" carrega um jogo salvo anteriormente e "Sair" fecha o programa.

4.2 Tela de jogo

A tela de jogo aparece quando "Novo jogo" ou "Carregar jogo" forem selecionados. Nela, é apresentado o tabuleiro de jogo, a pontuação, a quantidade de movimentos, maiores pontuações e todos os comandos possíveis.



Os comandos devem apenas ser digitados pelo usuário e o programa irá recebê-los e executá-los. Para qualquer direcional digitado, o tabuleiro realiza o movimento referente ao direcional digitado. Os botões "N" e "S" geram uma caixa na tela com a confirmação da execução do comando, enquanto o botão "ESC" informa que o jogo foi encerrado e retorna para o menu inicial. O botão "BACKSPACE" desfaz a jogada e volta o tabuleiro para a posição anterior.

Quando as condições para o fim de jogo forem atendidas, uma caixa aparece informando que o jogo foi encerrado. Caso a pontuação tenha sido maior que as últimas dez maiores, o programa solicita que o usuário digite um nome para salvar a pontuação e retorna para o menu principal.