Programação I
2019/2020

Universidade de Évora

Curso: Engenharia Informática

Projeto: 2048

Trabalho realizado por:
André Rato – 45517
Afonso Alves – 45286

Introdução:

Este trabalho foi realizado no âmbito da cadeira de Programação I a pedido da professora Teresa Gonçalves e tem como objetivo a criação de uma biblioteca de funções que permita a criação do jogo 2048 em dois modos, um modo iterativo e um modo automático. A linguagem utilizada neste trabalho é C.

Este relatório consiste na explicação detalhada do programa, apresentação das funções e variáveis utilizadas ao longo do mesmo.

Breve resumo:

O 2048 é um jogo de raciocínio criado pelo desenvolvedor italiano Gabriele Cirulli em março de 2014, em que o objetivo é deslizar peças numeradas com potências de base 2 numa grelha, combiná-las, tentando criar assim um azulejo com o número 2048.

Para o desenvolvimento do modo iterativo, é pedido ao utilizador o tamanho (N) do tabuleiro de jogo. Após isso é gerado o tabuleiro de dimensões N*N e com algarismos 2 ou 4 em duas posições aleatórias. Iterativamente, o jogador escolhe um sentido (baixo – B, cima – C, direita – D, esquerda – E), sendo apresentada no ecrã a grelha atualizada, juntamente com um novo algarismo (2 ou 4) que é gerado numa posição aleatória após todos os movimentos relativos a essa jogada. O jogo termina quando não é possível fazer mais nenhuma operação, quando existe uma peça com o valor de 2048 no tabuleiro ou quando o utilizador escolhe F. Assim que o jogo termina é apresentado o número total de peças combinadas e o número de peças de cada número na grelha final.

Já no caso do modo automático a grelha inicial está totalmente preenchida, sendo toda a informação lida de um ficheiro. O ficheiro contém uma linha com o tamanho do tabuleiro (N), N linhas com N algarismos cada uma e uma linha com as jogadas. No final são apresentados no ecrã o número total de peças combinadas e o número de peças restantes no tabuleiro de cada número.

Funções utilizadas no programa:

Função:	Argumentos e retornos:
random_de_array(*numeros, numero_elementos): esta função escolhe uma posição aleatória num determinado conjunto de valores, como um array.	**Argumentos:** *numeros – conjunto de valores de onde será escolhida a posição; tipo inteiro. numero_elementos – indica o "subtamanho" do array; tipo inteiro. **Valor de retorno:** numeros[pos] – retorna a posição dos elementos escolhidos no array; tipo inteiro.
possivel_colocar(tamanho, grelha): esta função verifica se é possível gerar, ou não, uma peça nova no tabuleiro.	**Argumentos:** tamanho – tamanho inserido pelo utilizador no início do jogo; tipo inteiro. grelha[tamanho][tamanho] – array de arrays de dimensões tamanho*tamanho associado ao tabuleiro de jogo; tipo inteiro. **Valor de retorno:** possíveis – retorna o número de valores 0 que existem na matriz; tipo inteiro.
colocar_numero(tamanho, grelha): esta função encontra uma linha e uma coluna aleatoriamente até que na posição da matriz seja encontrado um 0. Se encontrar, coloca uma nova peça no tabuleiro (2 ou 4, aleatoriamente).	**Argumentos:** tamanho – tamanho inserido pelo utilizador no início do jogo; tipo inteiro. grelha[tamanho][tamanho] – array de arrays de dimensões tamanho*tamanho associado ao tabuleiro de jogo; tipo inteiro.
possivel_jogar(tamanho, grelha): esta função verifica se é possível jogar, ou seja, se há peças iguais adjacentes ou se há peças com o valor 0.	**Argumentos:** tamanho – tamanho inserido pelo utilizador no início do jogo; tipo inteiro. grelha[tamanho][tamanho] – array de arrays de dimensões tamanho*tamanho associado ao tabuleiro de jogo; tipo inteiro. **Valor de retorno:** a função retorna 1 se encontrar peças iguais adjacentes, peças com o valor de 2048 ou peças com o valor de 0, e retorna 0 se não verificar nenhuma destas opções; tipo inteiro.
inicializar(tamanho, grelha, n_elementos): esta função gera a matriz de dimensões tamanho*tamanho com n elementos (n_elementos) diferentes de 0.	**Argumentos:** tamanho – tamanho inserido pelo utilizador no início do jogo; tipo inteiro. grelha[tamanho][tamanho] – array de arrays de dimensões tamanho*tamanho associado ao tabuleiro de jogo; tipo inteiro. n_elementos – número de elementos que serão gerados com os valores 2 ou 4.
mostrar(tamanho, grelha): esta função percorre todas as posições da matriz, dando print do seu valor; se o valor for 0, dá print de '-'.	**Argumentos:** tamanho – tamanho inserido pelo utilizador no início do jogo; tipo inteiro. grelha[tamanho][tamanho] – array de arrays de dimensões tamanho*tamanho associado ao tabuleiro de jogo; tipo inteiro.
contagem_pecas(tamanho, grelha): esta função faz a contagem das peças que estão no tabuleiro e apresenta o seu número, se diferente de 0.	**Argumentos:** tamanho – tamanho inserido pelo utilizador no início do jogo; tipo inteiro. grelha[tamanho][tamanho] – array de arrays de dimensões tamanho*tamanho associado ao tabuleiro de jogo; tipo inteiro.

direita(tamanho, grelha): esta função realiza o movimento para a direita; primeiramente, verifica horizontalmente se não há peças com o valor de 0 entre as outras cujo valor é diferente de 0; se houver coloca todos os valores diferentes de 0 à direita e os 0 à esquerda; depois, se houver peças iguais adjacentes horizontalmente, duplica o valor da peça da direita e a peça da esquerda fica com o valor 0; após isso volta a realizar o primeiro processo.	**Argumentos:** tamanho – tamanho inserido pelo utilizador no início do jogo; tipo inteiro. grelha[tamanho][tamanho] – array de arrays de dimensões tamanho*tamanho associado ao tabuleiro de jogo; tipo inteiro. **Valor de retorno:** contador – retorna o número de combinações (somas) que foram realizadas no movimento; tipo inteiro.
esquerda(tamanho, grelha): esta função realiza o movimento para a esquerda; primeiramente, verifica horizontalmente se não há peças com o valor de 0 entre as outras cujo valor é diferente de 0; se houver coloca todos os valores diferentes de 0 à esquerda e os 0 à direita; depois, se houver peças iguais adjacentes horizontalmente, duplica o valor da peça da esquerda e a peça da direita fica com o valor 0; após isso volta a realizar o primeiro processo.	**Argumentos:** tamanho – tamanho inserido pelo utilizador no início do jogo; tipo inteiro. grelha[tamanho][tamanho] – array de arrays de dimensões tamanho*tamanho associado ao tabuleiro de jogo; tipo inteiro. **Valor de retorno:** contador – retorna o número de combinações (somas) que foram realizadas no movimento; tipo inteiro.
cima(tamanho, grelha): esta função realiza o movimento para cima; primeiramente, verifica verticalmente se não há peças com o valor de 0 entre as outras cujo valor é diferente de 0; se houver coloca todos os valores diferentes de 0 em cima e os 0 em baixo; depois, se houver peças iguais adjacentes verticalmente, duplica o valor da peça de cima e a peça de baixo fica com o valor 0; após isso volta a realizar o primeiro processo.	**Argumentos:** tamanho – tamanho inserido pelo utilizador no início do jogo; tipo inteiro. grelha[tamanho][tamanho] – array de arrays de dimensões tamanho*tamanho associado ao tabuleiro de jogo; tipo inteiro. **Valor de retorno:** contador – retorna o número de combinações (somas) que foram realizadas no movimento; tipo inteiro.
baixo(tamanho, grelha): esta função realiza o movimento para baixo; primeiramente, verifica verticalmente se não há peças com o valor de 0 entre as outras cujo valor é diferente de 0; se houver coloca todos os valores diferentes de 0 em baixo e os 0 em cima; depois, se houver peças iguais adjacentes verticalmente, duplica o valor da peça de baixo e a peça de cima fica com o valor 0; após isso volta a realizar o primeiro processo.	**Argumentos:** tamanho – tamanho inserido pelo utilizador no início do jogo; tipo inteiro. grelha[tamanho][tamanho] – array de arrays de dimensões tamanho*tamanho associado ao tabuleiro de jogo; tipo inteiro. **Valor de retorno:** contador – retorna o número de combinações (somas) que foram realizadas no movimento; tipo inteiro.
jogada(): esta função pede ao utilizador uma letra (D, E, C, B ou F – direita, esquerda, cima, baixo ou fim, respetivamente); após isso, verifica a validade da jogada (se a letra inserida corresponde ou não às letras indicadas anteriormente); se a resposta não for válida volta a pedir ao utilizador uma nova letra até que esta corresponda a uma das letras pré-definidas.	**Valor de retorno:** resp – retorna a letra correspondente à jogada; tipo char.
Numero(): esta função pede ao utilizador o tamanho do tabuleiro de jogo e avalia se este está compreendido entre 2 e 10; se o valor não estiver compreendido entre 2 e 10, a função volta a pedir ao utilizador o tamanho do tabuleiro de jogo, até que este esteja compreendido entre os valores definidos.	**Valor de retorno:** tamanho – retorna o tamanho introduzido pelo utilizador; tipo inteiro.

Variáveis utilizadas nos programas:

- Modo Iterativo:

int tamanho – tamanho do tabuleiro de jogo;
int mov=0 – número de movimentos de cada jogada;
int pontos=0 – número total de movimentos;
int grelha[tamanho][tamanho] – tabuleiro de jogo;
char resp – caractere que define a jogada.

- Modo Automático:

int tamanho – tamanho do tabuleiro de jogo;
int mov=0 – número de movimentos de cada jogada;
int pontos=0 – número total de movimentos;
int grelha[tamanho][tamanho] – tabuleiro de jogo;
int numero – variável utilizada para associar os números do ficheiro às posições da matriz;
char result – variável utilizada para ler os caracteres do ficheiro.

Descrição do programa:

- Modo Iterativo:

O programa começa por chamar a função numero(), igualando-a à variável tamanho, de tipo inteiro. Nesta função é pedido ao utilizador o tamanho do tabuleiro de jogo.

Após isso, o tabuleiro é criado com as dimensões tamanho*tamanho e com dois elementos (2 ou 4) gerados aleatoriamente com a função inicializar(tamanho, grelha, 2).

De seguida, a função jogada() é chamada e o retorno da mesma é associado à variável resp de tipo char. Nesta função é pedido o caractere correspondente à primeira jogada realizada no jogo.

Depois disto, o programa entra num ciclo while que acaba quando não for possível jogar, ou seja, quando a função possivel_jogar(tamanho,grelha) retornar um valor diferente de 1, ou quando o caractere fornecido pelo utilizador for 'F', ou seja, quando a função jogada() retornar a variável resp associada ao caractere 'F'. Este ciclo é composto por um conjunto de condições correspondentes aos movimentos. A primeira instrução if corresponde ao movimento para a direita, a segunda ao movimento para a esquerda, a terceira para o movimento para cima e a quarta para o movimento para baixo. Em cada instrução if relacionada com movimentos é associada à variável inteira mov o valor retornado pela função correspondente ao movimento desejado; após isto é chamada a função colocar_numero(tamanho, grelha) onde o programa coloca um número (2 ou 4) numa posição aleatória da matriz que seja 0. Depois da colocação da nova peça, se possível, o programa junta o número de movimentos realizados nessa jogada ao número de movimentos totais realizados (variável pontos de tipo inteiro).

Após o movimento, o tabuleiro é mostrado no ecrã já com alterações realizadas através da função mostrar(tamanho, grelha).

Antes de retornar ao início do ciclo while é lida a última instrução if onde se avalia a existência de movimentos possíveis. Se existirem movimentos possíveis a função jogada() é chamada de novo.

Quando uma das proposições presentes na condição do ciclo while for falsa, o ciclo termina e é apresentado no ecrã o número total de combinações (pontos) e é feita a contagem das peças restantes no tabuleiro através da função contagem_pecas(tamanho, grelha).

- Modo Automático:

O programa é iniciado com a identificação das variáveis e com a abertura no modo de leitura (“r”) do ficheiro da qual serão lidas as informações do jogo (neste caso o ficheiro a ser lido é “jogadas.txt”).

Após a abertura do ficheiro, é analisado se a abertura do ficheiro foi realizado com sucesso ou não. Se o ficheiro não for aberto aparecerá a mensagem “Ficheiro inválido” e o programa irá terminar. Se o ficheiro for aberto aparecerá a mensagem “Ficheiro aberto com sucesso”.

Feita a abertura do ficheiro, é lido o primeiro caractere do ficheiro, através a função getc(ficheiro), e gravado na variável result (tipo char), que corresponde ao tamanho do tabuleiro.

Feita a leitura do primeiro caractere, é associada à variável tamanho o valor de result após convertido de char para int através da função atoi. De seguida é mostrado no ecrã o tamanho lido.

Depois, é criada a grelha de dimensões tamanho*tamanho através da função inicializar(tamanho, grelha, 0), em que todas as posições serão ocupadas com o valor 0. No passo seguinte é lido o caractere seguinte do ficheiro e o programa entra num conjunto de dois ciclos for: o primeiro que percorre as linhas da grelha e o segundo que percorre as colunas da grelha. Dentro do ciclo while é testado se o caractere lido é ‘2’ ou ‘4’. Se não for, será lido outro caractere até que este seja ‘2’ ou ‘4’. Assim que este satisfaça a condição referida anteriormente, o caractere armazenado na variável result é associado à variável inteira numero, logo após a sua conversão novamente com recurso à função atoi. Dada a conversão, a variável numero é associada à posição da grelha em que os ciclos estão a correr. De seguida lê-se outro caractere. Este processo repete-se até que a grelha esteja completa.

Assim que a grelha está completa, o programa lê outro caractere e entra num ciclo while que acaba quando é atingido o fim do ficheiro, quando a função possivel_jogar(tamanho, grelha) retorna um valor diferente de 1 ou quando o caractere lido é ‘F’.

Dentro deste ciclo, se o caractere lido e armazenado em result for ‘D’ é realizado o movimento para a direita, se for ‘E’ é realizado para a esquerda, se for ‘C’ realiza para cima e se for ‘B’ realiza para baixo. Em cada movimento, é guardado na variável mov o valor que é retornado das funções correspondentes ao movimento realizado e após isso esse valor é somado ao número total de combinações realizadas (pontos).

Após sair do ciclo while, e à semelhança do modo iterativo, o programa apresenta no ecrã o número total de combinações (pontos) assim como a contagem das peças, através da função contagem_pecas(tamanho, grelha).

Por fim é necessário fechar o ficheiro aberto inicialmente, através do comando fclose(ficheiro).

Conclusão e comentários críticos:

Foram realizados vários testes para encontrar a melhor opção para gerar a matriz com duas posições aleatórias com os algarismos 2 ou 4, mas tais retornavam sempre a mesma hipótese de escolha de posições. Por exemplo, quando utilizado o tamanho=4 eram colocadas duas peças na mesma posição sempre que se iniciava o programa. Deste modo foram realizadas várias tentativas ao programa para verificar que a colocação dos tais algarismos era feita aleatoriamente.

Foram realizados também vários testes relacionados aos movimentos pois era necessário que, para cada movimento, as posições da grelha fossem avaliadas em sentido contrário ao do movimento. Por exemplo, se fosse escolhido o movimento para a direita (da esquerda para a direita), a função teria de avaliar da direita para a esquerda. É de notar também que se, por exemplo, o tamanho=4, a posição de maior índice terá índice 3 (tamanho-1).

Verificou-se que ocorriam erros na leitura do tamanho. Quando inserido um caractere não inteiro no tamanho, o programa entra num ciclo while infinito.

É de observar também que, quando pedido um caractere para a jogada (B, C, E ou D) e fossem inseridos dois caracteres, o programa lê os dois caracteres e executa a jogada correspondente. Se, por exemplo, a jogada escolhida for “BD”, o programa irá fazer o movimento para baixo, e depois o movimento para cima. Se, por exemplo, a jogada for “BG”, o programa realiza o movimento para baixo e alerta o utilizador para uma jogada inválida.

Relacionados ao modo automático foram também realizados vários testes em relação a qual função utilizar para ler caractere a caractere o ficheiro. Poderiam ter sido escolhidas outras funções para esta leitura, tais como a função fscanf() ou a função fgets().