UNIVERSIDADE DE ÉVORA

Programação I - 2048 - Trabalho prático - Relatório

Trabalho realizado por:

Miguel Correia nº 15919

Tomás Dias nº42784

Introdução

Foi proposto na cadeira de **Programação I** o desenvolvimento de uma aplicação, em linguagem **C,** para jogar *2048*.

Todo o funcionamento do jogo encontra-se no enunciado disponibilizado para a realização deste relatório.

Com as informações do enunciado estudadas e bem interpretadas, prosseguiu-se para o desenvolvimento da aplicação.

Desenvolvimento

Antes de qualquer procedimento prático, foi necessário perceber quais funções iriam ser criadas. Assim, definiu-se que teriam de ser obrigatoriamente implementadas as seguintes **funções**:

- int baixo();
- int cima();
- int direita();
- int esquerda();
- int jogada();
- void mostrar().

Com isto, procurou-se a funcionalidade destas.

Função: int baixo(int grelha[][], int sz)

Esta função recebeu como **argumentos**:

- **int grelha[][]**: variável que representa a matriz correspondente à estrutura do jogo com a dita dimensão;
- **int sz**: variável que representa o tamanho da matriz quadrada, isto é, indica o número de linhas e colunas da mesma;

Esta função do tipo **int** tem como principal objetivo atualizar a grelha de jogo, de modo a que os valores não nulos da mesma se direcionem para baixo até à base da grelha ou até que outro valor não nulo se encontre abaixo destes.

Para o efeito, primeiramente é necessário percorrer todos os elementos da grelha, coluna por coluna de cima para baixo.

Se encontrar um 0 (espaço vazio) irá verificar se o elemento em cima é diferente de 0, se isso se verificar, então o a posição do zero toma o valor da posição que está em cima da posição do zero e esta o valor de 0. No entanto, se verificar que o valor da posição que está a ser avaliada é igual ao valor da linha de cima, então o valor da posição que está a ser avaliada toma o valor correspondente à soma dele mesmo com o valor da linha de cima enquanto que esta toma o valor 0.

Nesta função também é criada a variável **int count,** na qual vai ser guardado o número de vezes que uma peça da grelha foi movida. Posteriormente, será verificado se o valor desta variável será maior que 0, e se se verificar, a função chama a função **random_spot_num(),** responsável por gerar um novo valor não nulo numa posição aleatória da grelha.

Por fim, também é declarada a variável **num_pecascomb** responsável por guardar o número de peças combinadas. Ela é incrementada se for verificado que o valor da posição que está a ser avaliada é igual ao valor da linha de cima. Esta variável é devolvida no fim pela função **int baixo()**.

Função int cima(int grelha[][], int sz)

Esta função recebeu como **argumentos**:

- **int grelha[][]**: variável que representa a matriz correspondente à estrutura do jogo com a dita dimensão;
- **int sz**: variável que representa o tamanho da matriz quadrada, isto é, indica o número de linhas e colunas da mesma;

Esta função do tipo **int** tem como principal objetivo atualizar a grelha de jogo, de modo a que os valores não nulos da mesma se direcionem para cima até à parte superior da grelha ou até que outro valor não nulo se encontre acima destes.

Para o efeito, primeiramente é necessário percorrer todos os elementos da grelha, coluna por coluna de cima para baixo.

Se encontrar um 0 (espaço vazio) irá verificar se o elemento em baixo é diferente de 0, se isso se verificar, então o a posição do zero toma o valor da posição que está em baixo da posição do zero e esta o valor de 0. No entanto, se verificar que o valor da posição que está a ser avaliada é igual ao valor da linha de baixo, então o valor da posição que está a ser avaliada toma o valor correspondente à soma dele mesmo com o valor da linha de baixo enquanto que esta toma o valor 0. Foi usada uma variável **aux** de auxílio.

Nesta função também é criada a variável **int count,** na qual vai ser guardado o número de vezes que uma peça da grelha foi movida. Posteriormente, será verificado se o valor desta variável será maior que 0, e se se verificar, a função chama a função **random_spot_num(),** responsável por gerar um novo valor não nulo numa posição aleatória da grelha.

Por fim, também é declarada a variável **num_pecascomb** responsável por guardar o número de peças combinadas. Ela é incrementada se for verificado que o valor da posição que está a ser avaliada é igual ao valor da linha de baixo. Esta variável é devolvida no fim pela função **int cima()**.

Função: int direita(int tabuleiro[][], int sz)

Esta função tem como **argumentos**:

- **int grelha[][]**: variável que representa a matriz correspondente à estrutura do jogo com a dita dimensão;
- **int sz**: variável que representa o tamanho da matriz quadrada, isto é, indica o número de linhas e colunas da mesma.

Esta função do tipo **int** tem como principal objetivo atualizar a grelha de jogo, de modo a que os valores não nulos da mesma se direcionem para a direita até à coluna mais à direita da grelha ou até que outro valor não nulo se encontre à direita destes.

Para o efeito, primeiramente é necessário percorrer todos os elementos da grelha, coluna por coluna de cima para baixo.

Se encontrar um 0 (espaço vazio) irá verificar se o elemento à esquerda é diferente de 0, se isso se verificar, então o a posição do zero toma o valor da posição que está a ser verificada e esta o valor de 0. No entanto, se verificar que o valor da posição que está a ser avaliada é igual ao valor da linha de baixo, então o valor da posição que está a ser avaliada toma o valor correspondente à soma dele mesmo com o valor da linha de baixo enquanto que esta toma o valor 0.

Nesta função também é criada a variável **int count,** na qual vai ser guardado o número de vezes que uma peça da grelha foi movida. Posteriormente, será verificado se o valor desta variável será maior que 0, e se se verificar, a função chama a função **random_spot_num(),** responsável por gerar um novo valor não nulo numa posição aleatória da grelha.

Por fim, também é declarada a variável **num_pecascomb** responsável por guardar o número de peças combinadas. Ela é incrementada se for verificado que o valor da posição que está a ser avaliada é igual ao valor da linha de baixo. Esta variável é devolvida no fim pela função **int cima()**.

Função: int jogada(int tabuleiro[][], int sz, int x, int y)

Esta função tem como **argumentos**:

- **int grelha[][]**: variável que representa a matriz correspondente à estrutura do jogo com a dita dimensão;
- **int sz**: variável que representa o tamanho da matriz quadrada, isto é, indica o número de linhas e colunas da mesma;

Esta função do tipo **void** tem como objetivo executar uma jogada. Para tal, começou-se por declar as variáveis **char input_direcao,** que irá guardar o input (letra correspondente a um dos sentidos) pedido ao utilizador, e a variável **int num_total_pecascomb** responsável por guardar o número total de peças combinadas (soma de cada variável **int num_pecascomb** de cada uma das funções correspondentes aos sentidos).

Nesta função, pede-se ao utilizador um input, sendo este correspondente a um determinado caso. Para cada caso, a função **int jogada()** chama uma determinada função. Se o input corresponder ao sentido baixo (letra B), a função chama **baixo()**, se o input corresponder ao sentido cima (letra C), a função chama **cima()**, se o input corresponder ao sentido direita (letra D), a função chama **direita(),** se o input corresponder ao sentido esquerda (letra E), a função chama **esquerda()** ou se o input corresponder ao fim do jogo (letra F), a função chama **exit().** Esta informação encontra-se contida num ciclo que irá parar quando o utilizador pretender sair do jogo (a função **exit()** é executada) ou quando o jogo finalizar.

Função: void mostrar(int tabuleiro[][], int sz)

Esta função tem como **argumentos**:

- **int tabuleiro[][]**: variável que representa a matriz correspondente à estrutura do jogo com a dita dimensão;
- **int sz**: variável que representa o tamanho da matriz quadrada, isto é, indica o número de linhas e colunas da mesma.

Esta função do tipo **void** é relativamente simples e tem como objetivo principal, permitir ao jogador o estado atual do tabuleiro e transformar todos os zeros em traços, dando a ideia de um espaço sem número.

É chamada no final de cada jogada quando todas as outras funções já tenham terminado de alterar estrutura do tabuleiro, expondo a disposição do mesmo após a jogada.

Para além das funções obrigatórias, foi implementada a seguinte **função**:

- void random_spot_num().

Função: void random_spot_num(int grelha[][], int sz)

Esta função tem como **argumentos**:

- **int grelha[][]**: variável que representa a matriz correspondente à estrutura do jogo com a dita dimensão;
- **int sz**: variável que representa o tamanho da matriz quadrada, isto é, indica o número de linhas e colunas da mesma.
- **int count**: variável que corresponde ao número de valores a serem gerados.

Esta função do tipo **void**, tem como objetivo gerar números em posições aleatórias da grelha. São utilizadas as variáveis **int value,** que serve como auxiliar, a variável **int coluna** e a variável **int linha**, que guardam valores aleatórios entre 0 e **sz** (com o auxílio da função **rand()**) e que vão corresponder ás coordenadas das posições da grelha, e a variável **int numero** que guarda aleatoriamente o valor 2 ou 4 (com o auxílio da função **rand()**), valor esse que vai ser colocado na grelha.

Modo interativo

Para este modo, foram utilizadas as seguintes variáveis na função **main()**:

- **int sz**: variável que representa o tamanho da matriz quadrada, isto é, indica o número de linhas e colunas da mesma.
- **int x** e **int y**: representam respetivamente o índice da linha e da coluna da matriz que constitui a grelha.
- **grelha:** é a matriz responsável pela estrutura da grelha de jogo.
- **sizeerror:** variável utilizada para verificar se o **sz** introduzido pelo utilizador se encontra entre os valores permitidos.

Nesta função, começou-se por pedir ao utilizador um input correspondente ao **sz** da **grelha**.

De seguida, criou-se a matriz correspondente à **grelha** com o auxílio da função **malloc().** Optou-se por utilizar esta função para zerar a matriz mais facilmente. Depois chamou-se a função **random_spot_num()** para gerar os 2 valores aleatórios (2 ou 4) e colocá-los numa posição aleatória da **grelha.**

Por fim, chamou-se a função **jogada().**

Erros neste modo

Alguns erros ocorreram no desenvolvimento deste modo, nomeadamente um erro associado à contagem do número de peças combinadas que por vezes é incorreta. Também não foi possível apresentar no ecrã de fim de jogo o número de peças de cada número na grelha final.