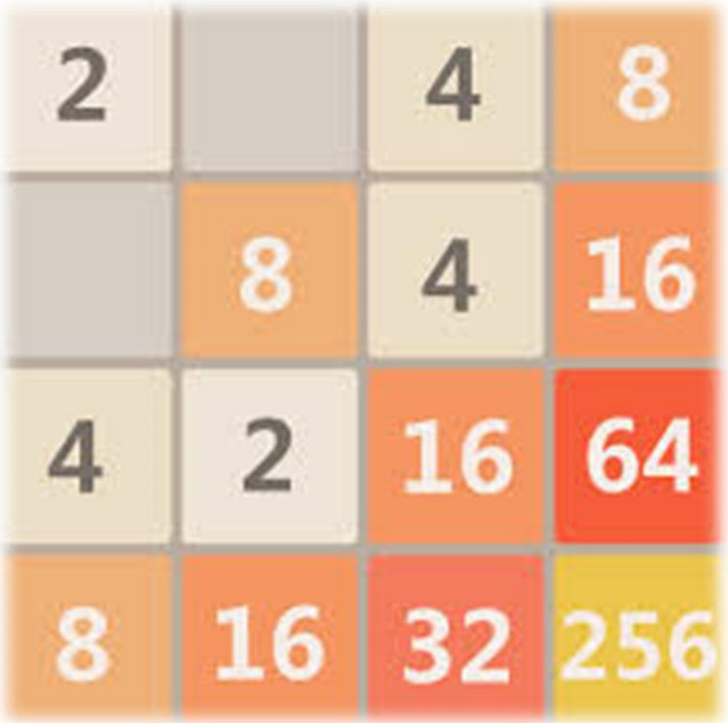
UNIVERSIDADE DE ÉVORA

2019/2020 – 3º Semestre

RELATÓRIO - TRABALHO PRÁTICO

PROGRAMAÇÃO I

2048



Trabalho realizado por:

Pedro Grilo, 43012

Diogo Castanho, 42496

1.1 - Introdução

O objetivo deste trabalho é programar um jogo já conhecido mundialmente e muito viciante, cujo nome é "2048", na linguagem *C.*

O jogo consiste num tabuleiro (vertical), cujo tamanho máximo será 10x10 (10 linhas e 10 colunas). *2048* é um jogo de quebra-cabeça para um jogador.

O objetivo do jogo é deslizar peças numeradas numa grelha de modo a combiná-las até criar uma peça com o número 2048.

O jogador poderá escolher um sentido em cada jogada de forma a juntar o máximo de peças com o mesmo número, sendo estas substituídas por uma nova peça cujo número é a soma das casas anteriormente juntas.

O jogo deverá apresentar 2 modos de jogo, um ***automático*** e um ***iterativo.***

Modo Iterativo

No modo iterativo a grelha inicial é preenchida em duas posições (aleatórias), com o número 2 ou 4.

Em cada jogada:

1. o jogador escolhe um sentido (B, C, D e E);

2. a grelha é atualizada de acordo com as regras do jogo;

3. é colocado um novo número 2 ou 4 numa posição vazia.

O jogo termina quando o utilizador escolhe F (Fim) ou quando não for possível combinar mais peças;

(Continuação 1.1)

Nessa altura é apresentado o número total de peças combinadas durante o jogo e a contagem do número de peças com cada número ainda no tabuleiro (por ordem crescente).

Este **modo** funciona da seguinte forma:

1. é solicitado ao utilizador o tamanho da grelha N;

2. é apresentada no ecrã uma grelha de tamanho N*N, com os algarismos 2 ou 4 em 2 posições (aleatórias);

3. iterativamente, o jogador escolhe um sentido, sendo apresentada no ecrã a grelha atualizada, mais um novo algarismo (2 ou 4) numa posição livre. O jogo termina quando não é possível fazer mais nenhuma operação ou quando o utilizador escolhe F.

4. nessa altura são apresentados o número total de peças combinadas e o número de peças de cada número na grelha final.

Modo Automático

No modo automático, a grelha inicial (totalmente preenchida) e as jogadas são lidas de um ficheiro de texto e o programa apresenta o número total de peças combinadas durante o jogo e a contagem após a última jogada.

(Continuação 1.1)

A **<u>informação de entrada</u>** contém:

- uma linha com o tamanho da grelha N;
- N linhas com N algarismos cada uma;
- uma linha com as K jogadas;

A resposta, a apresentar no ecrã, contém igualmente, o número total de peças combinadas o número de peças restantes (de cada número).

```
4
2 2 4 2
4 2 2 4
2 2 2 2
2 4 2 2
B D B E C
```

deverá ser apresentado no ecrã

```
pecas combinadas: 11
contagem: 0  2  2  1
```

1.2 - Decisões tomadas na realização do trabalho

Inicialmente, tentou perceber-se o problema em causa (neste caso, a programação de um jogo seguindo diversas regras impostas inicialmente) lendo mais que uma vez o enunciado proposto.

Após se perceber bem os objetivos do jogo, ter-se jogado um pouco do jogo online e observar o esqueleto das funções disponibilizadas (revendo o que cada uma deveria executar), o grupo deu início ao trabalho.

Assim sendo, optou-se começar por fazer o modo iterativo, simplesmente porque era o mais trabalhoso, difícil de implementar (na opinião do grupo) e cujo as funções seriam necessárias para o modo automático.

Após vários testes de algoritmo falhados, principalmente devido a erros de sintaxe (erros na digitação do código por vezes, por falta de concentração) e erros semânticos (onde o programa funcionava, porém, não fazia o pedido inicialmente), foi-se implementando o programa função a função.

Começou-se então por implementar individualmente cada uma das funções pedidas, utilizando desenhos feitos à mão para que fosse mais fácil perceber o objetivo de cada função e o que por exemplo cada sentido escolhido pelo utilizador iria causar no tabuleiro. As funções feitas foram depois utilizadas programa final ***("main").***

Uma vez que o grupo é constituído por 2 elementos inscritos pela 2ª vez na disciplina, houve outros trabalhos de 3º semestre que complicaram o tempo dedicado ao trabalho proposto este ano e por isso não se conseguiu implementar o modo automático.

Ainda assim houve um esforço enorme que permitiu que se conseguisse implementar totalmente e funcionalmente o **modo iterativo** do jogo proposto.

1.2.1 - Lista de funções utilizadas

- **Funções frequentes da biblioteca *stdio.h* (“printf”, “scanf”);**
- ***int rand ()*** (função pertencente à biblioteca ***(“stdlib.h”);***
- ***int baixo* (int grelha[ ][ ], int sz);**
- ***int cima* (int grelha[ ][ ], int sz);**
- ***int direita* (int grelha[ ][ ], int sz);**
- ***int esquerda* (int grelha[ ][ ], int sz);**
- ***int jogada* (int grelha[ ][ ], int sz, char sentido, int jogo**);**
- ***void mostrar* (int grelha[ ][ ], int sz);**
- ***void tabuleiro* (int grelha[ ][ ], int sz);**
- ***void imprime_contagem* (int grelha [ ][ ], int sz);**
- ***int verifica* (int grelha[ ][ ], int sz);**
- ***void adiciona* (int grelha[ ][ ], int sz);**

** **foi introduzido 1 parâmetro na função int jogada (int grelha[ ][ ], int sz, char sentido, int jogo) para o bom funcionamento do jogo**

1.3 - Desenvolvimento dos programas

Para além das funções básicas e frequentes da biblioteca stdio.h, tais como ***printf e scanf,*** foram implementadas outras funções necessárias para o bom funcionamento do jogo pedido tais como (baixo, cima, direita, esquerda, jogada, mostar, etc.).

Outra função utilizada neste jogo, foi a função rand () (pertencente à biblioteca ***stdlib.h***).

Esta função gera um número inteiro aleatório entre 0 e RAND_MAX e atribuirá um valor aleatório, neste caso foi utilizada para gerar 2 ou 4 funcionando para gerar aleatoriamente um tabuleiro inicial ou para adicionar aleatoriamente uma nova casa após uma jogada.

(1.3 - Continuação)

O trabalho foi realizado e organizado pela seguinte ordem:

1º

A primeira a ser implementada foi a função void mostrar (int grelha[][], int sz) sendo esta, responsável mostrar no ecrã a configuração atual da grelha.

A função void mostrar (int grelha[][], int sz), recebe a cada jogada como argumento, o estado atual da grelha e o tamanho da mesma, imprimindo-a e tornando-a visível para o utilizador.

Sendo uma função do tipo void não retorna qualquer valor.

2º

Posteriormente, foi implementada a função void tabuleiro (int grelha[ ][ ], int sz), tendo esta como objetivo gerar o tabuleiro inicial apresentando duas casas aleatórias com 2 ou 4.

Tem como argumentos a gralha inicial (composta apenas por 0´s) e o tamanho da mesma.

Utiliza a função rand () para que se consiga o objetivo proposto para a função.

Sendo uma função void não irá devolver nada.

(1.3 - Continuação)

3º

Após as funções de apresentação e configuração do tabuleiro, a terceira função implementada foi int baixo(int grelha[ ][ ], int sz), com o intuito de estabelecer o que o deveria acontecer caso o utilizador escolhesse o sentido B (baixo).

Esta função recebe como argumentos a grelha atual e o tamanho da mesma.

A função analisa a grelha a partir do canto inferior esquerdo e após a leitura da última posição da grelha e de feitas as alterações na grelha correspondentes à jogada selecionada retorna o número de peças combinadas na jogada.

4º

A quarta função implementada, com o objetivo de estabelecer o que o deveria acontecer caso o utilizador escolhesse o sentido C (cima) foi a função int cima(int grelha[ ][ ], int sz).

Esta função recebe como argumentos a grelha atual e o tamanho da mesma.

Tal e qual como a anterior analisa a grelha, porém a partir do canto superior esquerdo e após a leitura da última posição da grelha e de feitas as alterações na grelha correspondentes à jogada selecionada retorna o número de peças combinadas na jogada.

(1.3 - Continuação)

5º

Outra função do tipo int posteriormente implementada, foi a função int direita (int grelha[ ][ ], int sz), cujo os argumentos recebidos pela mesma são novamente a grelha atual e o tamanho da mesma e com o objetivo de estabelecer o que o deveria acontecer caso o utilizador escolhesse o sentido D (direita).

Esta analisa a grelha, a partir do canto inferior direito e após a leitura da última posição da grelha e de feitas as alterações na grelha correspondentes à jogada selecionada retorna o número de peças combinadas na jogada.

6º

Para terminar esta série de funções para o sentido selecionado, com o mesmo funcionamento das anteriores mas com o objetivo de estabelecer o que o deveria acontecer caso o utilizador escolhesse o sentido E (esquerda) fez-se a função int esquerda (int grelha[ ][ ], int sz).

Esta função recebe também como argumentos a grelha atual e o tamanho da mesma.

Da mesma forma que as três funções anteriores esta analisa a grelha a partir do canto inferior esquerdo e após a leitura da última posição da grelha e de feitas as alterações na grelha correspondentes à jogada selecionada retorna também o número de peças combinadas na jogada.

(1.3 - Continuação)

7º

Após as funções para cada sentido o grupo optou por fazer uma função à parte para o que aconteceria após cada jogada, isto é, teria que ser adicionado um novo algarismo (2 ou 4) numa posição livre da grelha.

Para isso criou-se a função void adiciona (int grelha[ ][ ], int sz), com esse mesmo objetivo.

Esta função recebe também como argumentos a grelha atual e o tamanho da mesma.

Esta, gera uma posição aleatória entre todas as posições da grelha onde o valor seja 0, ou seja, esteja vazia e adiciona um novo algarismo.

Sendo uma função void não irá devolver nada, porém faz as alterações necessárias na grelha para um bom funcionamento do jogo.

8º

Feitas as funções anteriores o grupo decidiu fazer a última função do "esqueleto" de funções que foi disponibilizado inicialmente.

A função int jogada (int grelha[ ][ ], int sz, char sentido, int jogo) faz a jogada do utilizador conforme o sentido que seja escolhido, utilizando as funções anteriormente feitas e adicionando as pecas combinadas de cada jogada a uma outra variável.

Esta, recebe como argumentos a grelha atual, o tamanho da mesma, o sentido selecionado pelo utilizador e um outro parâmetro adicionado pelo grupo (int jogo) para que enquanto o valor recebido nessa variável seja diferente de 1 o jogo continue e seja possível para o utilizador fazer outra jogada.

(1.3 - Continuação)

9º

A penúltima função implementada, também fora das funções inicialmente propostas foi a função int verifica (int grelha[ ][ ], int sz).

O grupo adicionou e implementou esta função ao programa com objetivo de verificar se existem ainda jogadas possíveis para utilizador poder continuar a jogar.

A função tem como argumentos a grelha atual e o tamanho da mesma (igualmente a algumas funções anteriores) e retorna o valor da variável int jogo sendo esta inicializada com valor 0 e alterado para 1 caso todas as condições impostas na função sejam falsas, ou seja, não haja mais hipóteses de jogada.

10º

Por fim, a última função implementada foi a função void imprime_contagem (int grelha [ ][ ], int sz).

Para terminar a implementação de todas as funções necessárias para o funcionamento do jogo, o grupo criou esta função com o objetivo de apresentar no final do jogo um dos pontos requeridos inicialmente:

- nº de peças de cada número na grelha final, ordenadas por ordem crescente.

Esta função recebe como argumentos mais uma vez, a grelha atual e o tamanho da mesma.

O funcionamento da mesma baseia-se em contar e imprimir um vetor onde foram postos os números contados das casas de cada numero que restam (ou ainda estão) no tabuleiro.

(1.3 - Continuação)

Foram usadas estas funções para o modo iterativo, sendo estas também as que o modo automático utilizaria. Além disso foram criados e utilizados os seguintes ficheiros:

- ***2048.h***, com os protótipos das funções comuns a ambos os programas;
- ***2048.c***, com a implementação das referidas funções;
- ***iterativo.c***, com o jogo para o modo iterativo;
- ***automatico.c***, com o jogo para o modo automático.