UNIVERSIDADE DE ÉVORA
CURSO DE ENGENHARIA INFORMÁTICA
PROGRAMAÇÃO I 2018/2019

COLOR SQUARES - RELATÓRIO

```
COLOR SQUARES

Regras: O tabuleiro de jogo e preenchido com numeros diferentes e o jogador pode re
mover grupos com numeros iguais, um grupo consiste num conjunto de quadrados com la
dos partilhados.
No final de cada jogada o tabuleiro e atualizado com as seguintes regras:
   gravidade: Os quadrados acima da  area vazia caem devido a gravidade.
   coluna: Quando toda a coluna esta vazia, esta colapsa movendo os blocos da direi
ta para a esquerda de modo a fechar a separacao.
```

Dinis Matos nº 42738

Jóse Santos nº 43017

Gerson Abreu nº 42501

Introdução

Após uma análise geral do trabalho proposto de Programação I (Color Squares), começou-se a criar as funções indicadas no trabalho para a execução de tal. As funções indicadas são marcar, pontuacao, gravidade, coluna, jogada e mostrar.

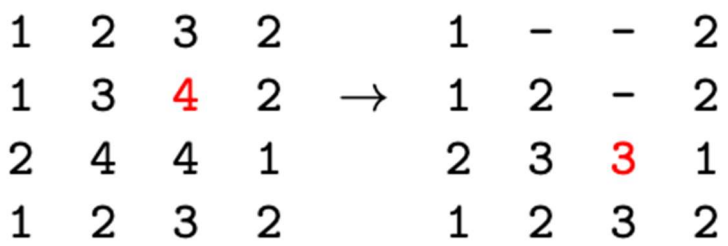
Função gravidade

O desempenho da função gravidade é importante para a realização do programa pois este permite a realização de tal. Esta função faz os números caírem se não houver números por baixo deles. Os seus argumentos são **int tabuleiro** e **int sz**.

Esta função irá verificar todos os números se são igual a zero, começando no canto superior esquerdo em seguida para baixo e depois faz o mesmo procedimento na coluna da direita e o mesmo se repete várias vezes. Se houver um número igual a zero, a função irá analisar todos os valores da coluna correspondentes e os "zeros" que existem nessa mesma coluna iram ficar em cima dos restantes números.

Dentro da função gravidade foram declaradas as variáveis **int x**, **int y** e **int a**. A variável **x**, irá verificar todas as colunas, "for (int x=0; x < sz; x++)", enquanto a variável **y** irá verificar todas as linhas, "for (int y=sz-1; y > 0; y--)". A variável **a** irá colocar os "zeros" no topo da respetiva coluna se, "if (tabuleiro[x][y]== 0)". Se sim, então "for (int a=1; a<sz; a++)" irá executar "if (tabuleiro[x][y-a]!=0 && y-a >= 0){ tabuleiro[x][y]=tabuleiro[x][y-a]; [x][y-a]= 0; a=sz;}".

Exemplo: (se retirar todos os "quatro" da matriz)



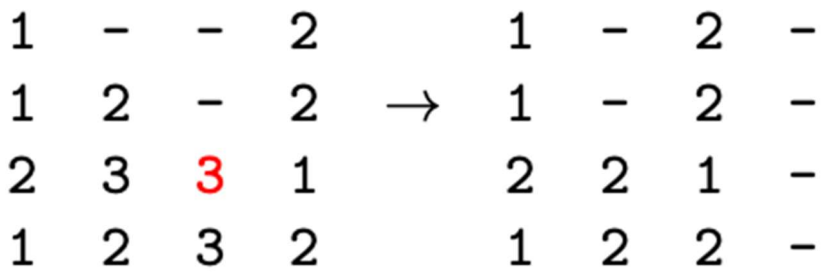
Função coluna

Tendo em conta a função gravidade, podemos afirmar que esta função também irá ser muito importante para o desempenho do programa. A função coluna irá colocar todas as colunas com pelo menos 1 número para a esquerda, fazendo com que todas as colunas que estejam à direita da ultima que tenham números sejam colunas apenas com "zeros". Esta função tem como argumentos **int tabuleiro** e **int sz**.

Esta função irá verificar se existe uma coluna só com "zeros", começando no canto inferior esquerdo em seguida para cima e depois faz o mesmo procedimento na coluna da direita e o mesmo se repete várias vezes. Se existir uma coluna só com "zeros", a função irá trocar essa coluna só com "zeros" com a coluna à direita. Para evitar erros na execução desta função, exemplo (duas colunas seguidas só com "zeros"), a função irá ser executada "**sz"** vezes para evitar esse tipo de erros.

Dentro da função coluna foram declaradas as variáveis **int x**, **int y**, **int i** e **int cont**. A variável **x**, irá verificar todas as colunas, "for (int x=0; x < sz; x++)", enquanto a variável **y** irá verificar todas as linhas, "for (int y=0; y<sz; y++)". A variável **cont** irá contar quantos zeros tem cada coluna e se "if (cont == sz)" essa coluna irá ser trocada com a coluna da direita, "for (int y=0; y<sz; y++){ tabuleiro [x][y] = tabuleiro [x+1][y]; tabuleiro [x+1][y]=0; }". A variável **i** servirá para executar "**sz**" vezes para evitar erros no programa.

Exemplo: (se retirar todos os "3" da matriz)



Função marcar

Esta função servirá para verificar todas as cores (x,y) que fazem parte do mesmo grupo, ou seja, se duas cores estão juntas uma à outra. Os argumentos desta função são **int tabuleiro, int sz, int x** e **int y**. A função cria duas variáveis, a **int n_quadrados**, que guarda o número de cores juntas, e a **int valor**, que guarda a posição de cada cor. Sendo assim, a função verifica todas as cores à esquerda, direita, cima e baixo de cada número (por exemplo, esta parte do código verifica se a cor é igual à de cima: "if(tabuleiro[x][y+1]==valor && y+1 < sz){n_quadrados += marcar(tabuleiro, sz, x, y+1)}"), e verificando sempre também se o número próximo ainda faz parte do tabuleiro ou se é maior ou igual a 0. No final, todo o grupo selecionado será substituído por "zeros" e a função irá dar "return" de **n_quadrados.**

Função pontuacao

Esta função tem como finalidade calcular a pontuação de uma jogada. A forma apresentada na função: "pontos = (numquadrados * (numquadrados + 1)) / 2" calcula o número de pontos de uma jogada. Neste caso, a função apenas tem um único argumento **int numquadrados**, em que este é recebido da função marcar e tem uma variável **int pontos** em que esta guarda o valor da pontuação da jogada. Por fim, esta função dá "return" da variável **pontos**.

Função jogada

No caso desta função, tem como fim, executar uma jogada. Os argumentos desta função são **int tabuleiro** e **int sz**. Dentro desta função estão incluidas as funções pontuacao, marcar, gravidade e coluna. Também foi criada a variável **int pont** para guardar a pontuação de cada jogada. Esta função da "return" da pontuação.

Função mostrar

Esta função é destinada a mostrar o estado atual do tabuleiro, tendo os argumentos **int tabuleiro** e **int sz,** começa por verificar todas as posições de (x,y), pois foram criadas as variáveis **int x** e **int y**, e caso alguma posição tenha o valor de 0, mostra um hífen(-) no lugar do 0 nessa mesma posição (“if(tabuleiro[x][y]==0){printf(" - ");}”). Caso o valor guardado nessa mesma posição não for igual a 0, mostra esse mesmo valor na respetiva posição (“else{printf(" %d ", tabuleiro[x][y]);}”). Para finalizar a função, mudamos de linha.

O trabalho consiste em desenvolver uma aplicação para jogar Color Squares. Existem 2 modos de jogo: o modo interativo e o modo automático.

Modo Interativo

Int sz – armazena o valor, escolhido pelo utilizador, para o tamanho do tabuleiro

Int x e **Int y** – armazenam as coordenadas introduzidas pelo utilizador, também utilizadas para construir o tabuleiro

Int tabuleiro [20][20] – Matriz responsável por guardar o tabuleiro de jogo

Int errosize – Verifica se o sz introduzido pelo utilizador está entre 1 e 20

Int pont – Armazena os pontos de cada jogada, e no final devole o resultado

Int errozero – Verifica se as coordenadas x e y introduzidas pelo utilizador correspondem a um valor que é “zero” ou estão fora no alcance (sz-1).

Ao executar o csIter, o programa começa por dar print do nome do jogo e das regras. É pedido ao utilizador o valor de “sz”, um ciclo while(errosize =1) verifica se o valor de “sz” é maior que zero e menor que 21, se sim o ciclo é parado.

Depois é criado o tabuleiro, com auxilio da função “rand()”.

De seguida um ciclo while(tabuleiro[0][sz-1] != 0), este ciclo acaba quando todos os valores forem “zero”, é chamada a função mostrar, mais um ciclo while(errozero==1) que verifica se os valores de x e y são maiores que “0” e menores que “sz-1”, e que o valor de tabuleiro[x][y] não corresponde a um “zero”, se sim errosize=0, quebrando o ciclo.

É utilizado pont += jogada(tabuleiro, sz, x, y)

No final é utilizado o printf("Pontuacao Final: %d \n", pont) de modo a mostrar ao utilizador a sua pontuação final.

Modo Automático

Int a[500] – utilizado para armazenar os valores do tabuleiro de jogo e totalmoves, copiados do ficheiro de texto

Int tabuleiro [20][20] – Matriz responsável por guardar o tabuleiro de jogo

Int sz – armazena o valor do tamanho do tabuleiro

Int totalmoves – armazena o número de jogadas introduzidas

Int x e **Int y** – armazenam as coordenadas introduzidas pelo utilizador, também utilizadas para construir o tabuleiro

Char t1[10000] – utilizado para guardar as coordenadas introduzidas pelo jogador, copiadas do ficheiro de texto

Int i – Utilizada para percorrer a matriz a[500] e o array t1[10000]

Int b, Int k e **int j** – variáveis auxiliares no cálculo do número de caracteres a ler do ficheiro, das coordenadas inseridas pelo utilizador.

Int pont - Armazena os pontos de cada jogada, e no final devole o resultado

Char nomef – armazena o nome do ficheiro de texto que o utilizador pretende analisar.

Ao executar o csAuto, o programa pede ao utilizador o nome do ficheiro para este ser lido. O programa começa por abrir o ficheiro e usando a função fscanf(), lê o número da primeira linha, e este é atribuído à variável sz. Depois um ciclo for() percorre o ficheiro, copiando os valores para o array a[500], sz*sz vezes, de modo a copiar a matriz toda. Logo a seguir outro ciclo for() que coloca os valores de a[500] na matriz tabuleiro[20][20]. Depois utilizando outro fscanf(), é atribuido o valor recebido à variável totalmoves. Um ciclo for calcula o valor de b, através de k, em que o valor de b aumenta 2 em quanto o k aumenta 1, até o valor de k ser igual ao valor de totalmoves, depois através da equação "j=(totalmoves*3)+b" conseguimos o valor de caracteres a ser copiados.

As coordenadas do tabuleiro são lidas através fscanf() e param no caracter j, e são colocadas no array t1[10000]

A seguir os valores de x e y são copiados a partir do array t1[10000], de modo a não copiar as virgulas. O y é invertido. E se o valor de tabuleiro[x][y] for diferente de "zero" a variável pont é atualizada, e a função jogada(tabuleiro, sz, x, y) é chamada.

No fim é utilizado o printf() para mostrar no ecrã o resultado final.