Programação I

Licenciatura em Engenharia Informática

Daniela Schmidt, João Sequeira, Rúben Teimas

Ano letivo 2023/24

1 Introdução

O *Ouri* é um jogo de 2 jogadores, constituído por um tabuleiro com 14 buracos e 48 pedras. O objetivo deste jogo é recolher mais pedras que o adversário. Todas as pedras têm o mesmo valor e vence o jogador que obtiver 25 ou mais pedras.

O tabuleiro é composto por duas filas de seis buracos, aos quais damos o nome de casas, e dois buracos maiores nas extremidades designados por depósitos.

Os jogadores sentam-se frente a frente, sendo as suas casas as que se encontram diretamente à sua frente, numeradas de 1 a 6, sendo a casa número 1 a casa mais à esquerda. Ao jogador pertence o depósito que se encontra à sua direita.

Para efeitos de simplicidade começa sempre a jogar o jogador do lado A (Fig.1).

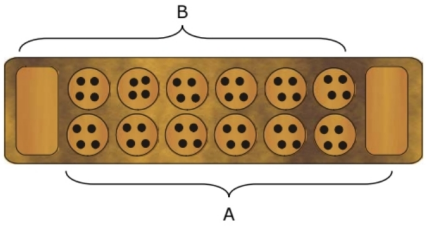

Figura 1: Estado inicial.

2 Regras

2.1 Movimentos

No início de cada jogo são colocados 4 pedras em cada uma das doze casas.

O jogador que abre o jogo colhe todas as pedras de uma das suas casas e distribui as pedras, uma a uma, nos buracos seguintes no sentido anti-horário (inverso ao dos ponteiros do relógio). Esta regra mantém-se para todas as jogadas.

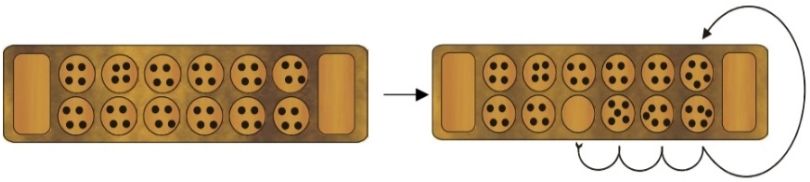

Figura 2: Distribuição de pedras numa jogada.

Se o jogador tiver mais do que 12 pedras, o jogador dá uma volta completa ao tabuleiro e salta a casa de onde partiu.

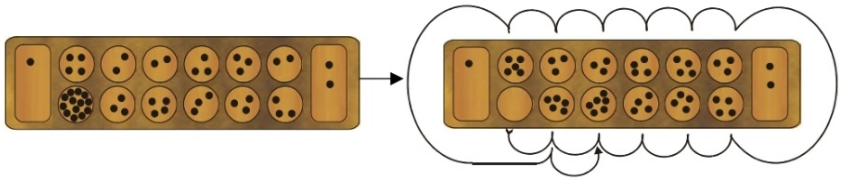

Figura 3: Distribuição de pedras, dando a volta ao tabuleiro.

Os jogadores não podem mexer em casas que contenham apenas uma pedra enquanto tiver casas com mais pedras.

2.2 Capturas

A captura é a última parte do movimento. Se ao depositarmos a última pedra numa casa do adversário e esta contenha apenas 2 ou 3 pedras (incluindo a que foi colocada por nós), podemos captura-las. Isto é, remover todas as pedras da casa e colocá-las no nosso depósito.

Sempre que as casas anteriores consecutivas tenham 2 ou 3 pedras, e pertençam ao adversário, devemos captura-las até que encontremos uma casa que não cumpra uma destas condições.

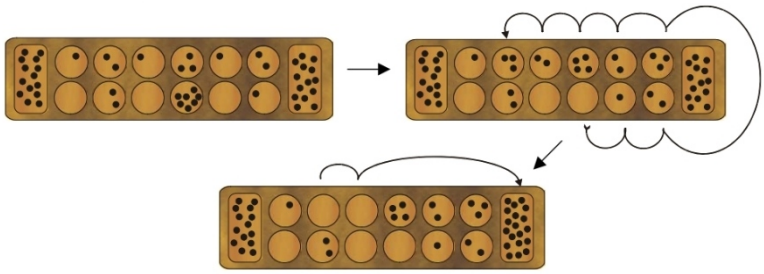

Figura 4: Jogada com captura.

Qualquer situação que não respeite as condições anteriores não permite fazer uma captura.

2.3 Regras complementares

Se ao realizar um movimento o jogador ficar sem pedras no seu lado do tabuleiro, o adversário é obrigado a efetuar um movimento que introduza pedras do seu lado.

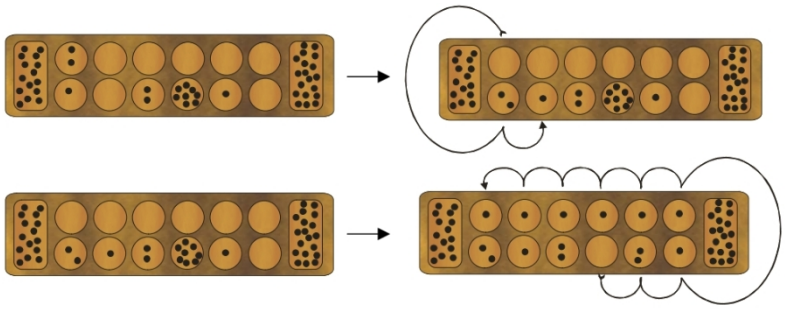

Figura 5: Adversário repõe as pedras no lado contrário.

Uma situação semelhante verifica-se caso o jogador capture todas as pedras do adversário, deixando-o sem possibilidade de jogar. Nesse caso deve ser o jogador a jogar de forma a introduzir pedras nas

casas do adversário.

Caso não seja possível, a partida termina (ver condições de término em 2.4).

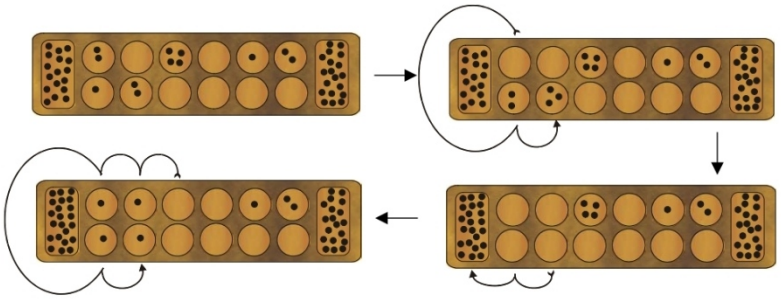

Figura 6: Repor as pedras no lado do adversário.

2.4 Fim da partida

Quando um jogador capturar a maioria das pedras (25 ou mais) a partida finaliza e esse jogador ganha.

Quando um jogador fica sem pedras e o adversário não consegue jogar de forma a colocar novamente pedras nas casas do jogador, a partida termina e o adversário recolhe as pedras dos seus buracos para o seu depósito. Vence o jogador que tiver mais pedras no seu depósito.

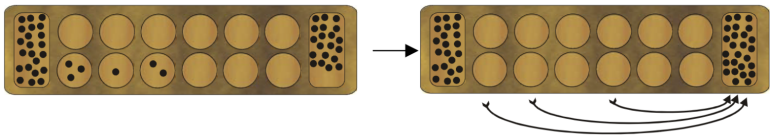

Figura 7: Situação impossivel de repor as pedras no lado do adversário.

Em alguns casos, perto do final, o jogo pode entrar num estado cíclico. Quando ocorrer esses casos, os jogadores devem recolher as pedras existentes nas suas casas e colocá-las nos respectivos depósitos.

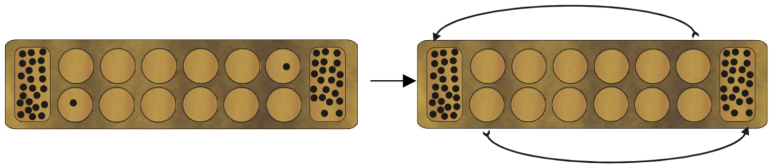

Figura 8: Situação de loop no estado do jogo.

3 Objetivo

Implementar, usando a linguagem C, um programa que simule um jogo de *Ouri*. O programa deve permitir dois modos de jogo:

- Jogador *VS* Jogador.
- Jogador *VS* Computador.

As jogadas do computador não necessitam de ser ótimas (ou sub-ótimas) mas devem respeitar as regras do jogo. Dotar o computador de "inteligência" (fazendo uma jogada ótima) será apenas valorizado

após assegurar a dinâmica do jogo: representar um tabuleiro, verificar a validade de uma jogada, alterar o tabuleiro de acordo com uma jogada válida, etc...

O programa deve perguntar ao jogador se quer jogar contra outro jogador ou contra o computador.

O estado do tabuleiro deve ser impresso na consola no início do jogo e após cada jogada e deve ter um formato semelhante a:

```
|---|--|--|--|--|--|--|---|
|   | 9| 0| 0| 9| 1| 2|   |
|  3|-----------------|  6|
|   | 3| 4| 9| 0| 1| 1|   |
|---|--|--|--|--|--|--|---|
```

O programa deve pedir ao jogador um input entre 0 e 6. Sendo 1 o buraco mais à esquerda do jogador e 6 o buraco mais à direita. O input 0 é usado para guardar o estado do tabuleiro num ficheiro. Ao escolher esta opção o programa deve pedir adicionalmente o nome do ficheiro e a execução deve ser terminada. O tabuleiro acima, ao ser guardado num ficheiro, deve ter o seguinte formato.

```
3
9 0 0 9 1 2
6
3 4 9 0 1 1
```

No final do programa, existindo um vencerdor, deve-se imprimir na consola qual o jogador que venceu e o número de pedras que capturou.

O compilador do código fonte do projeto deve resultar num ficheiro `ouri`. Para correr o jogo, deverá executar um dos seguintes comandos:

- ≫ `./ouri`
- ≫ `./ouri tabuleiro.txt`

Com o primeiro comando o tabuleiro é inicializado de acordo com as regras de inicio do jogo, enquanto que com o segundo comando o tabuleiro é inicializado com o conteúdo do ficheiro.

4 Submissão

A submissão do trabalho deve ser feita, num único ficheiro comprimido (`.zip` ou `.tar.gz`), através do moodle. O ficheiro deve conter:

- o código fonte (`.c` e `.h`).
- o relatório, em formato pdf.

Não inclua o executável resultante da compilação do programa. Para comprimir o ficheiro pode usar o seguinte comando no diretório onde tem o trabalho e relatório:

≫ `tar -czvf xxxxx_yyyyy_zzzzz.tar.gz *.c *.h *.pdf`

Em que `xxxxx`, `yyyyy` e `zzzzz` os respetivos números de aluno ordenados por ordem crescente.

O relatório deve descrever sucintamente a abordagem ao problema, dificuldades encontradas e possiveis soluções. Procure descrever as funções criadas durante o trabalho, isto é: o seu propósito, argumentos e valor de retorno.

5 Notas Finais

- O trabalho deverá ser efetuado por grupos de 2 a 3 elementos e entregue até à data indicada no moodle.
- O trabalho será discutido em dia e horário a anunciar.

- O trabalho será classificado de acordo com os seguintes fatores de avaliação: clareza dos algoritmos implementados, correção do código, legibilidade do código e qualidade do relatório.
- Será aplicado o código de conduta do Departamento de Informática. Em caso de fraude, para além reprovação à disciplina, a situação será reportada.
- Esclarecimentos adicionais poderão ser disponibilizados ao longo da execução do trabalho.

Bom trabalho!

① Introdução

```
#include   <stdio.h>
#include   <stdlib.h>

#define FILAS 1
#define CASAS 5
#define DEPOSITOS 1
#define PEDRAS_INICIAIS 4
```

PLAYER 0

12-4-4

N

7 <= x <= 12

CASA JOGADA + PEDRAS

4 → 13 % 12

3 → 14 % 12

6 - (12 - (5 + 4)) = 3

NCASAS — TOTAL CASAS — CASA JOGADA — PEDRAS