The War of the Five Kingdoms (±3)

Introdução à Programação
Enunciado do 2º Trabalho Prático, versão 1.1 – 2017-12-01
`Mensagens de comandos modificadas são apresentadas a verde`

Notas importantes

Prazo de entrega até às 17h00 do dia 15 de Dezembro de 2017.

Material a entregar neste trabalho REALIZADO EM GRUPOS DE 2 ALUNOS

- **Mooshak**: submissão e aceitação do código fonte pelo sistema de avaliação automática Mooshak. Consulte na página da disciplina as regras de submissão de trabalhos ao Mooshak.

Recomendações: A boa documentação do seu código fonte será valorizada, tal como a utilização do melhor estilo de programação possível, para além, claro, do correcto funcionamento do projecto. Não se esqueça de comentar as declarações das classes e dos seus métodos com uma explicação do seu significado, colocando pré-condições sempre que se justifique.

1 Introdução

Este documento descreve o 2º trabalho prático da disciplina de Introdução à Programação do 1º ano do Mestrado Integrado em Engenharia Informática. Descreve ainda uma metodologia de trabalho que os alunos devem seguir para desenvolver o referido trabalho.

O trabalho deve ser realizado **em grupos de 2 alunos**, de preferência do mesmo turno. Os alunos podem e devem tirar todas as dúvidas com os docentes da disciplina, bem como conversar com os colegas e debater soluções - mas nunca partilhando código - sempre respeitando os princípios do Código de Ética (disponível na página da cadeira no Moodle).

2 Desenvolvimento da Aplicação *The War of the Five Kingdoms (±3)*

2.1 Descrição do Problema

Pretende-se construir um jogo de estratégia baseado em turnos. Este jogo é disputado por um número variando entre **dois** e **oito** *reinos*. O objectivo de cada *reino* é eliminar os outros, enquanto evita ser eliminado por eles. Cada *reino* tem um *nome* e pode ter vários *castelos* e um *exército* de *soldados*. O jogo termina quando já só existe um *reino*. Para existir, um *reino* tem de ter, pelo menos, um *castelo* ou um *soldado* no seu *exército*. Cada *castelo* tem um *nome*, uma *localização* e um *tesouro*. O *castelo* pode pertencer a um *reino*, ou estar "abandonado". Quer pertença a um reino, quer esteja "abandonado", o *castelo* tem sempre um tesouro – uma determinada quantidade de moedas. Em cada turno, o *castelo* gera mais uma *moeda* para o seu

tesouro. Os *castelos* podem ser usados para recrutar *soldados* para o *exército* do *reino* que os detém, desde que o *tesouro* do *castelo* seja suficiente. Existem 3 tipos de *soldados* para recrutar, cada um com o seu *custo* associado: *cavaleiro* (4 moedas), *lanceiro* (2 moedas) e *espadachim* (2 moedas). O *exército* de um *reino* pode ter um número potencialmente ilimitado de *soldados*. Na prática, nunca ultrapassam o número de posições disponíveis no mapa de jogo. O *mapa de jogo* pode ser visto como um plano em que todas as coordenadas são dadas por valores naturais positivos.

Para iniciar um novo jogo, começa-se por estabelecer as dimensões do mapa, o número de reinos e o número de castelos existentes. Por exemplo, podem-se estabelecer uma dimensão *horizontal* de *10* e uma dimensão *vertical* de *20*, dando origem a um mapa com *10* x *20* posições (ou seja, em que as coordenadas horizontais válidas variam entre 1 e 10 e as verticais entre 1 e 20) - o canto inferior esquerdo do mapa tem sempre as coordenadas (1, 1).

Como se disse, o número de reinos é entre 2 e 8, logo o número de castelos é, no mínimo, um por reino, no máximo, um por posição no mapa. Por exemplo, no tal mapa de *10* x *20* posições, se existirem 3 reinos, podem ser criados entre 3 e 200 castelos. Em cada posição, apenas pode existir um castelo.

Uma vez indicado o número de reinos e castelos, serão criados e inseridos no jogo os castelos e finalmente, adicionam-se os reinos ao jogo. Cada reino é colocado num castelo que esteja "abandonado", passando a deter esse castelo e começa sem um exército. **O início de um novo jogo implica sempre o cancelamento do jogo anterior, se estiver algum a decorrer.**

Os reinos jogam alternadamente, pela ordem com que foram criados. A primeira jogada é do primeiro reino, a segunda do segundo, e assim sucessivamente, alternando entre os reinos que ainda não estiverem destruídos, até que sobre apenas um reino. Em cada jogada, um reino pode fazer as consultas que bem entender, mas apenas pode dar um comando passível de modificar o estado do jogo. Existem apenas dois comandos que modificam o estado do jogo: *recrutar* um soldado e *mover* um *soldado*.

Os soldados são recrutados (criados, na verdade) num castelo do reino, com um determinado custo. Para recrutar um soldado, o castelo necessita de ter moedas suficientes para pagar o custo de recrutamento e estar livre, ou seja, não pode ter nenhum soldado no seu interior. O soldado recrutado fica, automaticamente, dentro do castelo, ou seja, na mesma posição que o castelo. Para recrutar outro soldado nesse castelo, o reino terá primeiro de dar ordens para o soldado sair do castelo e depois então pode recrutar o novo soldado.

Em cada posição apenas pode existir um soldado. Sempre que um soldado se tenta deslocar para uma posição ocupada por um adversário, dá-se um combate "mortal". O soldado vencedor continua "vivo" e fica nessa posição. O derrotado "morre" e deixa de estar disponível para o jogo. As regras do combate são as seguintes:

- Os cavaleiros vencem sempre os espadachins.
- Os espadachins vencem sempre os lanceiros.
- Os lanceiros vencem sempre os cavaleiros.
- Quando dois soldados do mesmo tipo (e.g. dois cavaleiros) combatem, vence sempre o atacante, ou seja, o que se moveu.

Para movimentar um soldado, um jogador pode deslocar esse soldado para `norte`, `sul`, `este` ou `oeste`, uma posição de cada vez, não podendo ir ocupar a posição de outro soldado do mesmo reino. Os cavaleiros fazem 3 deslocamentos por jogada, enquanto que os espadachins e os lanceiros apenas fazem um deslocamento. Se durante um deslocamento um elemento do reino for para a posição de um castelo e conseguir entrar lá, conquista o castelo para o seu reino,

quer o castelo esteja "abandonado", quer pertença a outro reino. Note que o soldado pode ter que combater, caso o castelo esteja ocupado por um soldado de outro reino. Neste caso, só conquista o castelo se vencer o combate. O reino pode passar a recrutar soldados a partir desse castelo conquistado.

Se após um deslocamento (mesmo que seja um deslocamento intermédio, no caso do cavaleiro) o elemento encontrar um adversário, dá-se um combate em que um dos soldados "morre". Se após esse combate algum dos reinos ficar sem soldados e sem castelos, esse reino é destruído. O jogo termina quando já só existir um reino, sendo esse o reino vencedor.

No início do programa, este mostra os comandos disponíveis (tal como fará sempre que o utilizador escolher a opção `ajuda`). Quando se inicia a aplicação, o jogo não está activo. O utilizador tem de pedir para iniciar um novo jogo, através do comando `novo`. As opções `ajuda` e `sai` também devem estar disponíveis. Os restantes comandos apenas estão disponíveis durante o jogo.

Em cada jogada, o jogador escolhe um dos seus soldados indicando a sua localização e a direcção que o soldado deve usar para se deslocar: `norte`, `sul`, `este` ou `oeste`. Convencionase que ir para `norte` corresponde a incrementar a coordenada vertical da posição, para `sul` corresponde a decrementar essa coordenada, ir para `este` incrementa a coordenada horizontal, e ir para `oeste` decrementa a coordenada horizontal. Se o movimento for inválido, o que pode acontecer se não existir um soldado desse reino nessa posição, ou se o jogador tentar deslocar o soldado para fora do mapa, ou para uma casa ocupada por outro soldado do mesmo reino, o jogador perde esse movimento, sendo apresentada uma mensagem adequada. Note que um cavaleiro tem 3 movimentos. Mesmo que falhe algum movimento, desde que continue vivo, pode ainda usar os movimentos que lhe restem. Ou seja, se falhar, por exemplo, o primeiro movimento, continua a usar os dois seguintes. Os restantes soldados têm apenas um movimento por jogada. No final de cada movimento, pode ocorrer um combate. Nesse caso, apenas um dos combatentes sobrevive. Terminada a jogada, passa-se a vez ao adversário seguinte. Finalmente, se na sequência de uma jogada for derrotado um reino e apenas sobrar outro, o jogo termina. Os jogadores podem sempre começar um novo jogo.

A interação do utilizador com o programa é feita através de comandos, descritos na secção seguinte.

3 Comandos

Nesta secção apresentam-se os vários comandos que o sistema deve ser capaz de interpretar e executar. Nos exemplos apresentados, diferenciamos o `texto escrito pelo utilizador` da retroacção escrita pelo programa na consola, ao executar o comando. Pode assumir que o utilizador não cometerá erros na introdução de argumentos nos comandos, para além dos descritos neste enunciado, ou seja, apenas tem de tratar as situações de erro descritas aqui, pela ordem que são descritas.

Na leitura de comandos o interpretador não deve fazer distinção entre maiúsculas e minúsculas. Por exemplo, para o comando `sai`, o interpretador deve aceitar como válidas as alternativas `SAI`, `sai`, `Sai`, `sAI`, etc. Nos vários exemplos que se seguem, o símbolo ↵ denota a mudança de linha. Além disso, note também a utilização de uma *prompt*, denotada pelo símbolo >. Esta *prompt* indica que o programa aguarda um comando do utilizador. Portanto, **a *prompt* deve ser escrita na consola imediatamente antes da leitura de qualquer comando do utilizador**.

Caso o utilizador introduza um comando inexistente, o programa escreve na consola a mensagem Opcao inexistente.. Por exemplo o comando inexistente `desconhecido` teria este efeito:

```
> desconhecido↵
Opcao inexistente.↵
```

Vários comandos têm argumentos. Pode assumir que o utilizador apenas usa argumentos correctos. No entanto, pode acontecer que os argumentos passados a um desses comandos tenham algum valor incorrecto. Por esse motivo, teremos de testar esses argumentos exactamente pela ordem especificada neste enunciado.

3.1 Comando `sai`

Termina a execução do programa. O comando não necessita de argumentos.

```
> sai↵
Obrigado por jogar. Ate a proxima.↵
↵
```

Este comando tem sempre sucesso. Quando é invocado, o programa termina a sua execução, mesmo que isso interrompa um jogo.

3.2 Comando `ajuda`

Indica quais os comandos disponíveis no programa, atualmente. O comando não necessita de argumentos. O resultado do comando depende do contexto de utilização, sendo de considerar duas situações alternativas: quando o comando é invocado e não existe um jogo a decorrer, e quando o comando é invocado durante um jogo. Este comando tem sempre sucesso. No cenário seguinte, o comando é invocado num momento em que não está a decorrer nenhum jogo:

```
> ajuda↵
novo - Novo jogo↵
ajuda - Mostra a ajuda↵
sai - Termina a execucao do programa↵
```

No cenário seguinte, está a decorrer um jogo. Note que a *prompt* (nomeDoReino>) é diferente, quando há um jogo activo, indicando o nomeDoReino que está a jogar.

```
nomeDoReino > ajuda↵
novo - Novo jogo↵
soldado - Move o soldado↵
recruta - Recruta um soldado num castelo↵
mapa - Lista todos os castelos do mapa, incluindo os abandonados, pela ordem de criacao no jogo
e todos os reinos ainda em jogo, pela ordem de jogada↵
castelos - Lista os castelos do jogador activo, pela ordem pela qual foram conquistados↵
exercito - Lista os soldados vivos do jogador activo, pela ordem de recrutamento↵
reinos - Lista os varios reinos ainda em jogo, ordenados por nome do reino↵
ajuda - Mostra a ajuda↵
sai - Termina a execucao do programa↵
```

3.3 Comando `novo`

Inicia um novo jogo. Pode-se iniciar um `novo` jogo em qualquer momento. A simples tentativa de iniciar um jogo, mesmo que falhe, implica que se estiver a decorrer outro jogo, ele

é cancelado para se tentar começar um novo jogo.

Quando se inicia um novo jogo, definem-se as dimensões limite do mapa do jogo (largura e altura), que têm de ser inteiros não inferiores a 10. O mapa mais pequeno é de 10x10 posições, com canto inferior esquerdo na posição (1, 1). A seguir, indicam-se o número de reinos (um inteiro, no intervalo 2..8). Finalmente, indica-se o número de castelos a criar (um inteiro no intervalo [numeroDeReinos..largura*altura]. Se os dados introduzidos nesta linha tornarem impossível a inicialização correcta do jogo, o processo de inicialização de um jogo termina e não são pedidos ao utilizador mais dados de inicialização. Pelo contrário, se tudo estiver bem nesta linha inicial, segue-se um ciclo de leitura dos dados dos castelos e, se tudo continuar bem, outro ciclo, agora para ler os dados dos reinos. Para cada castelo, são dadas as coordenadas horizontal e vertical, a riqueza inicial (um número natural) e o nome do castelo (uma cadeia de caracteres que pode conter espaços e termina no fim da linha). Para cada reino é indicado o nome do reino (uma palavra, sem espaços) e o nome do castelo de origem do reino (tem de ser um dos castelos existentes). Quer os nomes dos castelos, quer os nomes dos reinos, devem ser lidos de modo a preservar as maiúsculas e minúsculas tal e qual como usadas pelo utilizador, no momento da criação (e.g. o castelos "Casterly Rock", "Winterfell" e "Castle Black" e os reinos "Lannister", "Stark" e "NightWatch"). Todas as coordenadas devem ser expressas por valores inteiros. O formato geral do comando é o seguinte (parâmetros com estilo **bold**):

```
> novo largura altura nReinos nCastelos↵
nCastelos castelos:↵
horizontal vertical riqueza nomeCastelo↵
horizontal vertical riqueza nomeCastelo↵
...↵
nReinos reinos:↵
reino nomeCastelo↵
reino nomeCastelo↵
...↵
```

Por exemplo, suponha que pretende iniciar um jogo com um mapa de 20x20, com reinos 3 reinos denominados de "Lannister", "Stark" e "NightWatch" e 4 castelos denominados "Winterfell", "Castle Black", "Casterly Rock" e "Dread Fort". Neste caso, tudo está correcto, sendo devolvida a mensagem (Jogo iniciado, comeca o reino Lannister.).

```
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Castle Black↵
Jogo iniciado, comeca o reino Lannister.↵
```

Caso algo esteja mal o jogo não é criado, mas se existia algum jogo a decorrer, ele é cancelado. Mostra-se uma mensagem de erro adequada:

- (1) Caso alguma das dimensões horizontal ou vertical seja inferior ao limiar mínimo (10), a mensagem de erro será (Mapa pequeno demais para o jogo.). Os parâmetros com o número de reinos e castelos têm de ser lidos, mas são ignorados. Como o comando falha logo na primeira linha, já não são pedidos os dados individuais sobre cada castelo e sobre cada

reino ao utilizador.

- (2) Se o número de reinos não respeitar os limites, a mensagem de erro apropriada será (Numero de reinos invalido.). O parâmetro com o número de castelos é ignorado. Como o comando falha logo na primeira linha, já não são pedidos os dados individuais sobre cada castelo e sobre cada reino ao utilizador.

- (3) Se o número de castelos não respeitar os limites, a mensagem de erro apropriada será (Numero de castelos invalido.). Como o comando falha logo na primeira linha, já não são pedidos os dados individuais sobre cada castelo e sobre cada reino ao utilizador.

- (4) Na colocação de um castelo, se o castelo for colocado fora do mapa, ou numa posição já ocupada por outro castelo, é apresentada uma mensagem de erro apropriada (Castelo em posicao invalida.). Se a riqueza for um valor inteiro não positivo, é apresentada uma mensagem de erro apropriada (Castelo com riqueza invalida.). Se o nome do castelo já existir, é apresentada uma mensagem de erro apropriada (Os castelos nao podem ter nomes duplicados.). Fica menos um castelo disponível no mapa, de cada vez que estes erros ocorrerem. Por exemplo, se em 5 castelos, 2 forem mal construídos, no final ficam 3 castelos disponíveis e a mensagem é escrita duas vezes, uma por cada castelo que foi impossível colocar.

- (5) Se o número de castelos colocados com sucesso for insuficiente, ou seja, se for inferior ao número de reinos pretendidos, é apresentada uma mensagem de erro apropriada (Numero insuficiente de castelos criados.) e a informação sobre os reinos não chega a ser pedida ao utilizador.

- (6) Na criação de um reino, se o nome do reino já existir, o reino não é criado e é apresentada uma mensagem de erro apropriada (Os reinos nao podem ter nomes duplicados.). Fica disponível menos um reino para o jogo, de cada vez que este erro ocorrer. Por exemplo, se entre 4 reinos houver um nome repetido 3 vezes, apenas serão criados 2 reinos, sem nomes repetidos. Apenas reinos que ainda não existam são criados.

- (7) Na criação de um reino, se o castelo pretendido já pertencer a outro reino, o reino não é criado e é apresentada uma mensagem de erro apropriada (Castelo ja ocupado.). Caso o castelo não existir, é apresentada uma mensagem de erro apropriada (Castelo nao existe.), e o reino não é criado. Fica disponível menos um reino para o jogo, de cada vez que estes erros ocorrerem. Por exemplo, se entre 4 reinos 3 deles tentarem escolher o mesmo castelo, apenas serão criados 2 reinos, sem castelos repetidos. Apenas reinos que ocupem castelos "abandonados" são criados.

- (8) Se o número de reinos criados com sucesso for menor que 2, o jogo não é criado e é apresentada uma mensagem de erro apropriada (Numero insuficiente de reinos criados.)

- (9) Os erros (1), (2), (3), (5) e (8) são suficientemente graves para abortar a criação de um jogo. Se qualquer deles ocorrer, deve ainda ser apresentada uma mensagem de erro apropriada (Erro fatal, jogo nao inicializado.). Note que os erros devem ser detectados e reportados pela ordem sugerida.

Seguem-se alguns exemplos ilustrativos da sequência de uma sessão interactiva em que ocorre um ou mais **erros** na criação de um novo jogo.

Erros (1), (9): Mapa pequeno demais para o jogo (3 exemplos)

```
> novo 3 16 3 4↵
Mapa pequeno demais para o jogo.↵
Erro fatal, jogo nao inicializado.↵
> novo 15 9 3 4↵
Mapa pequeno demais para o jogo.↵
Erro fatal, jogo nao inicializado.↵
> novo 5 6 3 4↵
Mapa pequeno demais para o jogo.↵
Erro fatal, jogo nao inicializado.↵
```

Erros (2), (9): Número de reinos inválido (2 exemplos):

```
> novo 10 10 1 4↵
Numero de reinos invalido.↵
Erro fatal, jogo nao inicializado.↵
> novo 10 10 9 4↵
Numero de reino invalido.↵
Erro fatal, jogo nao inicializado.↵
```

Erros (3), (9): Número de castelos inválido (2 exemplos):

```
> novo 10 10 4 2↵
Numero de castelos invalido.↵
Erro fatal, jogo nao inicializado.↵
> novo 10 10 4 123↵
Numero de castelos invalido.↵
Erro fatal, jogo nao inicializado.↵
```

Erro (4): Castelo em posição inválida ou Castelo com riqueza invalida ou Os castelos nao podem ter nomes duplicados (3 exemplos):

```
> novo 20 20 3 5↵
5 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
7 -2 8 Highgarden↵
Castelo em posicao invalida.↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Castle Black↵
Jogo iniciado, comeca o reino Lannister.↵
```

```
> novo 20 20 3 5↵
5 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
7 2 -8 Highgarden↵
Castelo com riqueza invalida.↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Castle Black↵
Jogo iniciado, comeca o reino Lannister.↵
```

```
> novo 20 20 3 5↵
5 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
7 2 4 Castle Black↵
Os castelos nao podem ter nomes duplicados.↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Castle Black↵
Jogo iniciado, comeca o reino Lannister.↵
```

Erros (5), (9): Número insuficiente de castelos criados:

```
> novo 20 20 3 5↵
5 castelos:↵
10 25 4 Winterfell↵
Castelo em posicao invalida.↵
12 20 1 Castle Black↵
0 3 10 Casterly Rock↵
Castelo em posicao invalida.↵
7 -2 8 Highgarden↵
Castelo em posicao invalida.↵
15 16 2 Dread Fort↵
Numero insuficiente de castelos criados.↵
Erro fatal, jogo nao inicializado.↵
```

Erro (6): Reinos com nomes repetidos:

```
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
Stark Castle Black↵
Os reinos nao podem ter nomes duplicados.↵
Jogo iniciado, comeca o reino Lannister.↵
```

Erro (7): Castelo ja ocupado ou Castelo nao existe (2 exemplos):

```
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
Targaryen Winterfell↵
Castelo ja ocupado.↵
Jogo iniciado, comeca o reino Lannister.↵
```

```
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
Targaryen Castle White↵
Castelo nao existe.↵
Jogo iniciado, comeca o reino Lannister.↵
```

Erros (8), (9): Número insuficiente de reinos criados:

```
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Winterfell↵
Stark Winterfell↵
Castelo ja ocupado.↵
Targaryen Winterfell↵
Castelo ja ocupado.↵
Numero insuficiente de reinos criados.↵
Erro fatal, jogo nao inicializado.↵
```

3.4 Comando **mapa**

Informa sobre o estado actual do jogo. Este comando apenas está disponível quando existe um jogo a decorrer. O comando não necessita de argumentos. Descreve o estado actual do jogo no que diz respeito à dimensão do mapa, os castelos no mapa e a ordem de jogadores/reinos a jogar. Note que ao contrário de no projecto 1, já não é suposto neste comando imprimir as posições e o estado de cada soldado. Essa descrição do jogo tem o seguinte formato:

reino > mapa↵
dimensaoHorizontal dimensaoVertical↵
nCastelos castelos:
nomeCastelo1 (**nomeDono1**)
nomeCastelo2 (**nomeDono2**)
...
nomeCasteloN (**nomeDonoN**)
mReinos reinos:
nomeReino1; **nomeReino2**; ... **nomeReinoM**

Onde **nomeDono** é o nome do reino a quem pertence o castelo, ou "sem dono", caso o castelo esteja abandonado. Por exemplo, suponha que foi criado o seguinte jogo:

```
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Castle Black↵
Jogo iniciado, comeca o reino Lannister.↵
```

O mapa seria apresentado assim:

```
Lannister > mapa↵
20 20↵
4 castelos:↵
Winterfell (Stark)↵
Castle Black (NightWatch)↵
Casterly Rock (Lannister)↵
Dread Fort (sem dono)↵
3 reinos:↵
Lannister; Stark; NightWatch↵
```

Neste segundo exemplo, nem todos os castelos e reinos são criados correctamente mas, apesar disso, estão reunidas as condições essenciais para começar o jogo.

```
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 160 2 Dread Fort↵
Castelo em posicao invalida.↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Winterfell↵
Castelo ja ocupado.↵
Jogo iniciado, comeca o reino Lannister.↵
```

O mapa seria apresentado assim:

```
Lannister > mapa↵
20 20↵
3 castelos:↵
Winterfell (Stark)↵
Castle Black (sem dono)↵
Casterly Rock (Lannister)↵
2 reinos:↵
Lannister; Stark↵
```

Este comando apenas está activo durante o jogo:

- (1) Caso não esteja a decorrer nenhum jogo, o programa deve escrever a mensagem (Comando inactivo.).

```
> mapa↵
Comando inactivo.↵
```

3.5 Comando `castelos`

Informa sobre os castelos do jogador activo (um reino). Este comando apenas está disponível quando existe um jogo a decorrer. O comando não necessita de argumentos. Lista o nome e a riqueza de todos os castelos do jogador activo, pela ordem pela qual foram conquistados ou ocupados pelo reino (inclui-se aqui, também, a ocupação inicial de um castelo por um reino, no início do jogo). Esta listagem tem o seguinte formato:

reino > castelos↵
nCastelosDoReino castelos:↵
nomeCastelo1 com riqueza **riquezaCastelo1** na posicao (**xCastelo1**,**yCastelo1**)↵
nomeCastelo2 com riqueza **riquezaCastelo2** na posicao (**xCastelo2**,**yCastelo2**)↵
...
nomeCasteloN com riqueza **riquezaCasteloN** na posicao (**xCasteloN**,**yCasteloN**)↵

Por exemplo, suponha que foi criado o seguinte jogo:

> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Castle Black↵
Jogo iniciado, comeca o reino Lannister.↵

O comando castelos seria apresentado assim:

Lannister > castelos↵
1 castelos:↵
Casterly Rock com riqueza 10 na posicao (1,3)↵

Suponha que, após algumas jogadas, o reino Lannister conquistou o castelo Castle Black com uma riqueza de 3, e que a riqueza actual de Casterly Rock é de 12. O comando castelos seria apresentado assim:

Lannister > castelos↵
2 castelos:↵
Casterly Rock com riqueza 12 na posicao (1,3)↵
Castle Black com riqueza 3 na posicao (12,20)↵

Caso o jogador activo não tenha castelos em seu poder, a mensagem a escrever na consola é (Sem castelos.), da seguinte forma:

Lannister > castelos↵
Sem castelos.↵

Este comando apenas está activo durante o jogo:

- (1) Caso não esteja a decorrer nenhum jogo, o programa deve escrever a mensagem (Comando inactivo.).

> castelos↵
Comando inactivo.↵

3.6 Comando `exercito`

Informa sobre o exército do jogador activo (um reino). Este comando apenas está disponível quando existe um jogo a decorrer. O comando não necessita de argumentos. Lista todos os soldados vivos do jogador activo, pela ordem pela qual foram recrutados pelo reino. Esta listagem tem o seguinte formato:

reino > `exercito`↵
nSoldadosDoReino soldados:↵
tipoDoSoldado1 na posicao (**xSoldado1**,**ySoldado1**)↵
tipoDoSoldado2 na posicao (**xSoldado2**,**ySoldado2**)↵
...
tipoDoSoldadoN na posicao (**xSoldadoN**,**ySoldadoN**)↵

Caso o jogador activo não tenha soldados no seu exército, a mensagem a escrever na consola é (Sem exercito.)

Por exemplo, suponha que foi criado o seguinte jogo:

> `novo 20 20 3 4`↵
4 castelos:↵
`10 15 4 Winterfell`↵
`12 20 1 Castle Black`↵
`1 3 10 Casterly Rock`↵
`15 16 2 Dread Fort`↵
3 reinos:↵
`Lannister Casterly Rock`↵
`Stark Winterfell`↵
`NightWatch Castle Black`↵
Jogo iniciado, comeca o reino Lannister.↵

O comando exercito executado imediatamente a seguir, seria apresentado assim:

Lannister > `exercito`↵
Sem exercito.↵

Suponha que, ao fim de algum tempo, o reino Lannister recrutou, sucessivamente, um cavaleiro, dois espadachins, um lanceiro e mais um cavaleiro. O comando `exercito` seria apresentado assim:

Lannister > `exercito`↵
5 soldados:↵
cavaleiro na posicao (4,3)↵
espadachim na posicao (3,3)↵
espadachim na posicao (2,2)↵
lanceiro na posicao (1,4)↵
cavaleiro na posicao (2,3)↵

Suponha que, após algumas jogadas, o primeiro espadachim e o lanceiro Lannister morreram, que os outros soldados se mantém exactamente no mesmo sítio, e que o reino Lannister recrutou mais um lanceiro. O comando `exercito` seria apresentado assim:

```
Lannister > exercito↵
4 soldados:↵
cavaleiro na posicao (4,3)↵
espadachim na posicao (2,2)↵
cavaleiro na posicao (2,3)↵
lanceiro na posicao (1,3)↵
```

Este comando apenas está activo durante o jogo:

- (1) Caso não esteja a decorrer nenhum jogo, o programa deve escrever a mensagem (Comando inactivo.).

```
> exercito↵
Comando inactivo.↵
```

3.7 Comando `reinos`

Lista os vários reinos ainda em jogo, ordenados por nome de reino. Este comando apenas está disponível quando existe um jogo a decorrer. O comando não necessita de argumentos. Lista todos os reinos ainda em jogo, por ordem alfabética. Para cada reino apresenta, numa linha, o nome, o número de castelos, o número de soldados e a riqueza total (o somatório das riquezas em cada um dos castelos do respectivo reino). Esta listagem tem o seguinte formato:

```
reino > reinos↵
nReinosActivos reinos:↵
nomeReino1, nCastelosReino1 castelos, nSoldadosReino1 soldados, nMoedasReino1 de riqueza↵
...
nomeReinoN, nCastelosReinoN castelos, nSoldadosReinoN soldados, nMoedasReinoN de riqueza↵
```

Por exemplo, suponha que foi criado o seguinte jogo:

```
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Castle Black↵
Jogo iniciado, comeca o reino Lannister.↵
```

O comando reinos executado imediatamente a seguir, seria apresentado assim:

```
Lannister > reinos↵
3 reinos:↵
Lannister, 1 castelos, 0 soldados, 10 de riqueza↵
NightWatch, 1 castelos, 0 soldados, 1 de riqueza↵
Stark, 1 castelos, 0 soldados, 4 de riqueza↵
```

Note que, sempre que existe um jogo activo existem necessariamente reinos activos. Este comando apenas está activo durante o jogo:

- (1) Caso não esteja a decorrer nenhum jogo, o programa deve escrever a mensagem (Comando inactivo.).

```
> reinos↵
Comando inactivo.↵
```

3.8 Comando **recruta**

Recruta um soldado num castelo. Os soldados são recrutados num castelo do reino desocupado (i.e., sem nenhum soldado lá dentro), com um determinado custo: *cavaleiro* (4 moedas), *lanceiro* (2 moedas) e *espadachim* (2 moedas). Este comando apenas está disponível quando existe um jogo a decorrer. O comando necessita de 2 argumentos: o tipo de soldado (cavaleiro, espadachim, ou lanceiro) e o nome do castelo. O comando tem o seguinte formato:

```
reino > recruta tipoSoldado nomeCastelo↵
```

Caso o castelo pertença ao jogador activo, tenha moedas suficientes e esteja livre, o recrutamento é realizado e o programa deve escrever a mensagem (**tipoSoldado** recrutado no **nomeCastelo** do reino **nomeReino** por **custoSoldado** moedas.). Neste exemplo, começamos por consultar os castelos do reino NightWatch e, depois, recrutamos um lanceiro em Castle Black.

```
NightWatch > castelos↵
1 castelos:↵
Castle Black com riqueza 6 na posicao (12,20)↵
NightWatch > recruta lanceiro Castle Black↵
lanceiro recrutado no Castle Black do reino NightWatch por 2 moedas.↵
```

No recrutamento de um soldado pode ocorrer as seguintes situações:

- (1) Caso não esteja a decorrer nenhum jogo, o programa deve escrever a mensagem (Comando inactivo.).
- (2) Caso o tipo de soldado não seja válido, o programa deve escrever a mensagem (Tipo de soldado inexistente.).
- (3) Caso o castelo não pertença ao jogador activo (um reino), o programa deve escrever a mensagem (Castelo invadido ilegalmente.).
- (4) Caso o castelo pertença ao jogador activo mas não tenha moedas suficientes, o programa deve escrever a mensagem (Riqueza insuficiente para recrutamento.).
- (5) Caso o castelo pertença ao jogador activo e tenha moedas suficientes, mas esteja ocupado com outro soldado, o programa deve escrever a mensagem (Castelo nao livre.).

Após a execução deste comando, a jogada termina e o jogador activo é actualizado, independentemente de se ter realizado com sucesso, ou não, o recrutamento de um soldado. Para além disso, todos os castelos geram uma moeda, por ser o fim de uma jogada.
Por exemplo, suponha que foi criado o seguinte jogo:

```
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 7 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Castle Black↵
Jogo iniciado, comeca o reino Lannister.↵
```

Seguem-se alguns exemplos dos erros que podem ocorrer com o comando `recruta`:
Erro (1): Comando inactivo (1 exemplo):

```
> recruta lanceiro Casterly Rock↵
Comando inactivo.↵
```

Erro (2): Tipo de soldado inexistente (1 exemplo): Neste exemplo, os Lannister tentam recrutar um `arqueiro`, que é um tipo de soldado não suportado pela aplicação. Ilustramos ainda que a tentativa, com ou sem sucesso, de recrutar um soldado consome uma jogada, passando-se o controlo do jogo para o reino seguinte (neste exemplo, os Stark).

```
Lannister > recruta arqueiro Casterly Rock↵
Tipo de soldado inexistente.↵
Stark > castelos↵
1 castelos:↵
Winterfell com riqueza 5 na posicao (10,15)↵
Stark >
```

Erro (3): Tentativa de recrutamento num castelo que não pertence ao reino (1 exemplo:)

```
Lannister > recruta cavaleiro Castle Black↵
Castelo invadido ilegalmente.↵
Stark > castelos↵
1 castelos:↵
Winterfell com riqueza 5 na posicao (10,15)↵
Stark >
```

Erro 4: Tentativa de recrutamento num castelo que já tem a sua guarnição preenchida com outro soldado (1 exemplo):

```
Lannister > recruta cavaleiro Casterly Rock↵
Castelo nao livre.↵
Stark > castelos↵
1 castelos:↵
Winterfell com riqueza 5 na posicao (10,15)↵
Stark >
```

Erros (1), (2), (3) e (4): este cenário envolve vários recrutamentos e vários tipos de erros (vários exemplos:)

```
novo - Novo jogo↵
ajuda - Mostra a ajuda↵
sai - Termina a execucao do programa↵
> recruta cavaleiro castelo↵
Comando inactivo.↵
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Castle Black↵
Jogo iniciado, comeca o reino Lannister.↵
Lannister > castelos↵
1 castelos:↵
Casterly Rock com riqueza 10 na posicao (1,3)↵
Lannister > mapa↵
20 20↵
4 castelos:↵
Winterfell (Stark)↵
Castle Black (NightWatch)↵
Casterly Rock (Lannister)↵
Dread Fort (sem dono)↵
3 reinos:↵
Lannister; Stark; NightWatch↵
Lannister > recruta soldado Casterly Rock↵
Tipo de soldado inexistente.↵
Stark > castelos↵
1 castelos:↵
Winterfell com riqueza 5 na posicao (10,15)↵
Stark > recruta cavaleiro Castle Black↵
Castelo invadido ilegalmente.↵
NightWatch > castelos↵
1 castelos:↵
Castle Black com riqueza 3 na posicao (12,20)↵
NightWatch > recruta cavaleiro Castle Black↵
Riqueza insuficiente para recrutamento.↵
Lannister > castelos↵
1 castelos:↵
Casterly Rock com riqueza 13 na posicao (1,3)↵
Lannister > recruta cavaleiro Casterly Rock↵
cavaleiro recrutado no Casterly Rock do reino Lannister por 4 moedas.↵
Stark > castelos↵
1 castelos:↵
Winterfell com riqueza 8 na posicao (10,15)↵
Stark > recruta espadachim Winterfell↵
espadachim recrutado no Winterfell do reino Stark por 2 moedas.↵
NightWatch > castelos↵
1 castelos:↵
Castle Black com riqueza 6 na posicao (12,20)↵
NightWatch > recruta lanceiro Castle Black↵
lanceiro recrutado no Castle Black do reino NightWatch por 2 moedas.↵
```

```
Lannister > castelos↵
1 castelos:↵
Casterly Rock com riqueza 12 na posicao (1,3)↵
Lannister > recruta lanceiro Casterly Rock↵
Castelo nao livre.↵
Stark > castelos↵
1 castelos:↵
Winterfell com riqueza 9 na posicao (10,15)↵
Stark > recruta espadachim Winterfell↵
Castelo nao livre.↵
NightWatch > castelos↵
1 castelos:↵
Castle Black com riqueza 7 na posicao (12,20)↵
NightWatch > recruta cavaleiro Castle Black↵
Castelo nao livre.↵
Lannister > mapa↵
20 20↵
4 castelos:↵
Winterfell (Stark)↵
Castle Black (NightWatch)↵
Casterly Rock (Lannister)↵
Dread Fort (sem dono)↵
3 reinos:↵
Lannister; Stark; NightWatch↵
Lannister > recruta soldado Casterly Rock↵
Tipo de soldado inexistente.↵
Stark > mapa↵
20 20↵
4 castelos:↵
Winterfell (Stark)↵
Castle Black (NightWatch)↵
Casterly Rock (Lannister)↵
Dread Fort (sem dono)↵
3 reinos:↵
Lannister; Stark; NightWatch↵
Stark > sai↵
Obrigado por jogar. Ate a proxima.↵
↵
```

3.9 Comando **soldado**

Movimenta um soldado. O comando necessita de obrigatoriamente três argumentos: coordenadas (abcissa e ordenada) onde se encontra o soldado a mover e uma direcção para indicar o movimento. Caso o soldado na posição dada seja um cavaleiro então deve existir mais dois argumentos (direcções de mais 2 movimentos). Lembre-se que os lanceiros e espadachins só podem fazer um movimento em cada jogada. No entanto, o cavaleiro pode executar três movimentos em cada jogada. O formato do comando é um dos seguintes:

```
reino > soldado xSoldado ySoldado direccao↵

reino > soldado xSoldado ySoldado direccao1 direccao2 direccao3↵
```

Cada **direccao** pode assumir os valores `norte`, `sul`, `este`, ou `oeste`. O primeiro formato, com uma direcção, é usado pelos soldados que se deslocam a pé (lanceiros e espadachins). O segundo formato, com 3 direcções, é usado pelos cavaleiros. Pode assumir que os testes no

Mooshak passam sempre um número de direcções adequado a cada soldado, desde que exista um soldado na posição em causa do reino que está a dar a ordem.

No exemplo que se segue, os três reinos ainda em jogo deslocam, sucessivamente, os seus soldados. Para facilitar a sua análise, na primeira jogada de cada reino é feita uma consulta ao exército respectivo.

```
Lannister > exercito↵
4 soldados:↵
cavaleiro na posicao (1,3)↵
espadachim na posicao (2,2)↵
cavaleiro na posicao (2,3)↵
lanceiro na posicao (1,1)↵
Lannister > soldado 1 3 norte este este↵
Lannister cavaleiro (1,4)↵
Lannister cavaleiro (2,4)↵
Lannister cavaleiro (3,4)↵
Stark > exercito↵
1 soldados: espadachim na posicao (10,15)↵
Stark > soldado 10 15 sul↵
Stark espadachim (10,14)↵
NightWatch > exercito↵
1 soldados: lanceiro na posicao (12,20)↵
NightWatch > soldado 12 20 oeste↵
NightWatch lanceiro (11,20)↵
Lannister > soldado 2 2 este↵
Lannister espadachim (3,2)↵
```

Este comando apenas está activo durante o jogo:

- (1) Caso não esteja a decorrer nenhum jogo, o programa deve escrever a mensagem (Comando inactivo.).

Se o jogo está a correr, o terreno de jogo é um local perigoso e várias coisas podem correr mal:

- (2) Na posição `(xSoldado,ySoldado)` não existe nenhum soldado. O programa deve escrever a mensagem (Nao existe nenhum soldado ilustre da casa de **reino** na posicao (**xSoldado**,**ySoldado**).).

- (3) O nosso valoroso soldado na posição `(xSoldado,ySoldado)` pode tentar deslocar-se para fora do mapa. Não toleramos desertores. Se o soldado tentar fugir do mapa, fica quietinho e ganha a fama de cobardolas sendo escrita a mensagem (O **tipoSoldado** da ilustre casa de **reino** e um cobardolas.).

- (4) Pode também acontecer que o soldado na posição `(xSoldado,ySoldado)` se tente deslocar para uma posição já ocupada por um elemento do seu reino que ainda esteja "vivo" (os soldados mortos não ocupam espaço no mapa, desaparecendo completamente do jogo ao serem derrotados). Quando a posição final de um movimento está ocupada por outro soldado do mesmo reino, o movimento deve falhar, ou seja, o soldado não se consegue mexer. Cada vez que se tentar deslocar para uma posição já ocupada deve ser escrita a mensagem (O **tipoSoldado** da ilustre casa de **reino** devia tentar ir para outro sitio.).

Além das situações de erro, há muitas outras situações que ocorrem no campo de batalha. Nomeadamente, ocorrem combates, conquistam-se castelos, etc.

- (A) Pode também acontecer que o soldado na posição `(xSoldado,ySoldado)` se tente deslocar para uma posição já ocupada por um castelo que está "livre" (nada impede que o soldado ocupe essa posição). Como a posição está livre, o movimento deve realizar-se, ou seja, o soldado mexe-se. Neste caso, deve ser escrita a mensagem (O **tipoSoldado** da ilustre casa de **reino** adquiriu um novo castelo **nomeCastelo** para o seu reino.).

- (B) Ao se deslocar, o soldado na posição `(xSoldado,ySoldado)` pode tentar invadir uma posição ocupada por um adversário. Neste caso, realiza-se um combate. Caso o soldado atacante vencer a batalha, o adversário "morre"e a mensagem a escrever é (Muhahah, sou um **reino**! Sou invencivel! Nenhum **tipoSoldadoRival** me faz frente!), em que **reino** representa o reino do soldado e **tipoSoldadoRival** o oponente derrotado em batalha.

- (C) No entanto, se o soldado na posição `(xSoldado,ySoldado)` é derrotado no combate "morre". A mensagem adequada, nesse caso, é (Argh! A dor! Maldito sejas, **tipoSoldadoRival reinoRival**.) em que **tipoSoldadoRival** é o tipo de soldado rival e **reinoRival** é o reino oponente.

- (D) Caso seja um cavaleiro, o soldado na posição `(xSoldado,ySoldado)`, pode acontecer que o soldado morra em um dos movimentos. Caso isso aconteça ~~- escrito na consola (O cavaleiro da ilustre casa de **reino** ja nao esta entre nos.), e~~ os seguintes movimentos são ignorados.

- (E) Na sequência de um combate, o soldado na posição `(xSoldado,ySoldado)` pode derrotar o adversário que se encontrava a proteger um dado castelo, adquirindo assim o castelo para o seu reino. Nesse caso, ele primeiro derrota o adversário (B) e depois conquista o castelo (A). Portanto, primeiro deve escrever a mensagem descrita em (B) e depois a descrita em (A).

Independente destas mensagens, no final de cada deslocação ou deslocação parcial (no caso do cavaleiro), deve reportar a nova posição do soldado ou a sua morte, com os seguintes formatos respectivamente:

reino tipoSoldado (**xSoldado**,**ySoldado**)↵

reino tipoSoldado morto↵

Seguem-se alguns exemplos de uma sessão interactiva em que ocorre um ou mais erros nas ordens dadas a um soldado.

Erro (1): Comando inactivo (não está a decorrer nenhum jogo):

```
> soldado 4 7 norte↵
Comando inactivo.↵
```

Erro (2): Não existe nenhum soldado do reino na posição indicada.

```
Stark > soldado 4 7 norte↵
Nao existe nenhum soldado ilustre da casa de Stark na posicao (4,7).↵
```

Erro (3): O soldado é um cobardolas. O soldado tenta-se mover para fora do mapa, sem sucesso.

Stark > `soldado 4 1 sul`↵
O lanceiro da ilustre casa de Stark e um cobardolas.↵
Stark lanceiro (4,1)↵

Neste exemplo num mapa de 20x20, o cavaleiro se encontra na posição (19, 12) e recebe a jogada este este norte. Nesse caso, ainda consegue fazer o primeiro movimento, mas falha o seguinte, conseguindo finalmente fazer o terceiro movimento:

Lannister > `soldado 19 12 este este norte`↵
Lannister cavaleiro (20,12)↵
O cavaleiro da ilustre casa de Lannister e um cobardolas.↵
Lannister cavaleiro (20,12)↵
Lannister cavaleiro (20,13)↵

Erro (4): Soldado tenta mover-se para posição já ocupada por outro soldado do mesmo reino, sem sucesso. São apresentados dois exemplos de soldados (um lanceiro e um cavaleiro) que, em determinado momento, tentam ocupar uma posição onde já se encontra outro soldado do mesmo reino.

Stark > `soldado 12 8 sul`↵
O lanceiro da ilustre casa de Stark devia tentar ir para outro sitio.↵
Stark lanceiro (12,8)↵
Targaryen > `soldado 14 18 sul sul este`↵
O cavaleiro da ilustre casa de Targaryen devia tentar ir para outro sitio.↵
Targaryen cavaleiro (14,18)↵
O cavaleiro da ilustre casa de Targaryen devia tentar ir para outro sitio.↵
Targaryen cavaleiro (14,18)↵
~~O cavaleiro da ilustre casa de Targaryen devia tentar ir para outro sitio.↵~~
Targaryen cavaleiro (15,18)↵

De seguida, apresentamos vários exemplos de combates e conquistas.
Combate (A): Soldado conquista um castelo.

Stark > `soldado 12 19 norte`↵
O espadachim da ilustre casa de Stark adquiriu um novo castelo Castle Black para o seu reino.↵

Combate (B): Soldado atacante derrota soldados defensores em combate. No exemplo que se segue, o `cavaleiro` do reino `Lannister` na posição (10, 10) tenta-se deslocar 3 vezes para `sul`, encontrando, sucessivamente, um cavaleiro Stark, um espadachim NightWatch e a terceira posição vazia. O `cavaleiro` derrota o cavaleiro e o espadachim adversários, pelo caminho. No entanto, na jogada seguinte, é morto por um `lanceiro Stark`.

Lannister > `soldado 10 10 sul sul sul`↵
Muhahah, sou um Lannister! Sou invencivel! Nenhum cavaleiro me faz frente!↵
Lannister cavaleiro (10,9)↵
Muhahah, sou um Lannister! Sou invencivel! Nenhum espadachim me faz frente!↵
Lannister cavaleiro (10,8)↵
Lannister cavaleiro (10,7)↵
Stark > `soldado 9 7 este`↵
Muhahah, sou um Stark! Sou invencivel! Nenhum cavaleiro me faz frente!↵
Stark lanceiro (10,7)↵

Combate (C): Soldado atacante morre em batalha.

Lannister > `soldado 10 10 sul`↵
Argh! A dor! Maldito sejas, espadachim Stark.↵
Lannister espadachim morto↵

Combate (D): Cavaleiro atacante morre em combate. Vamos ver o que acontece quando um cavaleiro morre no primeiro, segundo e terceiro movimentos de um comando soldado. Neste exemplo os três cavaleiros são mortos por lanceiros adversários. Naturalmente, um cavaleiro morto deixa de se mexer, de receber ordens, e desaparece do exército do qual fazia parte.

Lannister > `soldado 4 1 norte norte norte`↵
Lannister cavaleiro (4,2)↵
Argh! A dor! Maldito sejas, lanceiro Stark.↵
Lannister cavaleiro morto↵
Stark >`soldado 4 10 sul sul sul`↵
Stark cavaleiro (4,9)↵
Stark cavaleiro (4,8)↵
Argh! A dor! Maldito sejas, lanceiro Lannister.↵
Stark cavaleiro morto↵
NightWatch >`soldado 15 10 este este este`↵
NightWatch cavaleiro (16,10)↵
NightWatch cavaleiro (17,10)↵
NightWatch cavaleiro (18,10)↵
Argh! A dor! Maldito sejas, lanceiro Lannister.↵
NightWatch cavaleiro morto↵

Combate (C), (D): Soldado atacante derrota soldados defensores em combate, mas acaba por ser derrotado. No exemplo que se segue, o `cavaleiro` do reino `Lannister` na posição (10, 10) tenta-se deslocar 3 vezes para `sul`, encontrando, sucessivamente, um cavaleiro, um espadachim e um lanceiro adversários do reino `Stark`. O `cavaleiro` derrota um cavaleiro e um espadachim adversários, sendo finalmente derrotado por lanceiro.

Lannister > `soldado 10 10 sul sul sul`↵
Muhahah, sou um Lannister! Sou invencivel! Nenhum cavaleiro me faz frente!↵
Lannister cavaleiro (10,9)↵
Muhahah, sou um Lannister! Sou invencivel! Nenhum espadachim me faz frente!↵
Lannister cavaleiro (10,8)↵
Argh! A dor! Maldito sejas, lanceiro Stark.↵
Lannister cavaleiro morto↵

Combate (E): Soldado conquista castelo derrotando soldado defensor.

WhiteWalker > `soldado 13 20 oeste`↵
Muhahah, sou um WhiteWalker! Sou invencivel! Nenhum lanceiro me faz frente!↵
O espadachim da ilustre casa de WhiteWalker adquiriu um novo castelo Castle Black para o seu reino.↵
WhiteWalker espadachim (12,20)↵

Exemplo: Um exemplo dum cenário que envolve vários movimentos "Simples"

novo - Novo jogo↵
ajuda - Mostra a ajuda↵
sai - Termina a execucao do programa↵
> novo 20 20 3 4↵
4 castelos:↵
10 15 4 Winterfell↵
12 20 1 Castle Black↵
1 3 10 Casterly Rock↵
15 16 2 Dread Fort↵
3 reinos:↵
Lannister Casterly Rock↵
Stark Winterfell↵
NightWatch Castle Black↵
Jogo iniciado, comeca o reino Lannister.↵
Lannister > castelos↵
1 castelos:↵
Casterly Rock com riqueza 10 na posicao (1,3)↵
Lannister > recruta cavaleiro Casterly Rock↵
cavaleiro recrutado no Casterly Rock do reino Lannister por 4 moedas.↵
Stark > castelos↵
1 castelos:↵
Winterfell com riqueza 5 na posicao (10,15)↵
Stark > recruta lanceiro Winterfell↵
lanceiro recrutado no Winterfell do reino Stark por 2 moedas.↵
NightWatch > castelos↵
1 castelos:↵
Castle Black com riqueza 3 na posicao (12,20)↵
NightWatch > recruta espadachim Castle Black↵
espadachim recrutado no Castle Black do reino NightWatch por 2 moedas.↵
Lannister > exercito↵
1 soldados:↵
cavaleiro na posicao (1,3)↵
Lannister > soldado 1 3 norte norte norte↵
Lannister cavaleiro (1,4)↵
Lannister cavaleiro (1,5)↵
Lannister cavaleiro (1,6)↵
Stark > exercito↵
1 soldados:↵
lanceiro na posicao (10,15)↵
Stark > soldado 10 15 sul↵
Stark lanceiro (10,14)↵
NightWatch > exercito↵
1 soldados:↵
espadachim na posicao (12,20)↵

NightWatch > `soldado 12 20 oeste`↵
NightWatch espadachim (11,20)↵
Lannister > `exercito`↵
1 soldados:↵
cavaleiro na posicao (1,6)↵
Lannister > `castelos`↵
1 castelos:↵
Casterly Rock com riqueza 12 na posicao (1,3)↵
Lannister > `recruta lanceiro Casterly Rock`↵
lanceiro recrutado no Casterly Rock do reino Lannister por 2 moedas.↵
Stark > `exercito`↵
1 soldados:↵
lanceiro na posicao (10,14)↵
Stark > `castelos`↵
1 castelos:↵
Winterfell com riqueza 9 na posicao (10,15)↵
Stark > `recruta cavaleiro Winterfell`↵
cavaleiro recrutado no Winterfell do reino Stark por 4 moedas.↵
NightWatch > `exercito`↵
1 soldados:↵
espadachim na posicao (11,20)↵
NightWatch > `castelos`↵
1 castelos:↵
Castle Black com riqueza 7 na posicao (12,20)↵
NightWatch > `recruta cavaleiro Castle Black`↵
cavaleiro recrutado no Castle Black do reino NightWatch por 4 moedas.↵
Lannister > `castelos`↵
1 castelos:↵
Casterly Rock com riqueza 13 na posicao (1,3)↵
Lannister > `exercito`↵
2 soldados:↵
cavaleiro na posicao (1,6)↵
lanceiro na posicao (1,3)↵
Lannister > `soldado 1 6 norte norte este`↵
Lannister cavaleiro (1,7)↵
Lannister cavaleiro (1,8)↵
Lannister cavaleiro (2,8)↵
Stark > `exercito`↵
2 soldados:↵
lanceiro na posicao (10,14)↵
cavaleiro na posicao (10,15)↵
Stark > `soldado 10 15 este oeste este`↵
Stark cavaleiro (11,15)↵
Stark cavaleiro (10,15)↵
Stark cavaleiro (11,15)↵
NightWatch > `sai`↵
Obrigado por jogar. Ate a proxima.↵
↵

3.10 O fim de uma jogada

Existem apenas dois comandos capazes de fazer terminar uma jogada: `recruta` e `soldado`. No final da execução de uma jogada, caso o jogo não acabe, o jogador activo é actualizado, ou seja, passa a vez ao jogador seguinte. A mudança de jogador activo ocorre independentemente

do sucesso do comando de `recruta` ou `soldado`. No final da jogada, todos os castelos, quer pertençam a um reino, quer não, geram mais 1 moeda para o respectivo tesouro.

3.11 O fim do jogo

O jogo pode terminar na sequência de um movimento de um soldado porque:

- (1) O soldado na posição `(xSoldado,ySoldado)` conquista o último castelo do último adversário que já não tinha exército. Nesse caso, o reino do soldado que efectua a conquista vence o jogo.
- (2) Na sequência de um combate, o soldado na posição `(xSoldado,ySoldado)` pode derrotar o último adversário do último reino activo sem castelos, vencendo assim o jogo.
- (3) Se o soldado na posição `(xSoldado,ySoldado)` for derrotado e for o último do seu reino (já sem castelos), e só existirem 2 reinos activos, essa derrota resulta na perda do jogo.

No caso de fim do jogo, é escrita a mensagem (Sou um heroi **reino**! Vitoria gloriosa!), onde **reino** é o reino vitorioso.

Exemplos: No exemplo que se segue, existem apenas 2 reinos activos:`Lannister` e `Stark`. O `cavaleiro` do reino `Lannister`, na posição (10, 10) tenta-se deslocar 3 vezes para `sul`, encontrando no primeiro movimento um cavaleiro do reino `Stark`. O `cavaleiro` derrota o cavaleiro da reino `Stark`, que era o último sobrevivente desse reino, que também já não tinha castelos. Sem exército e sem castelos, o reino `Stark` está derrotado. Assim, o jogo está ganho pelo reino `Lannister`. Neste cenário, os dois últimos movimentos são ignorados.

```
Lannister > soldado 10 10 sul sul sul↵
Muhahah, sou um Lannister! Sou invencivel! Nenhum cavaleiro me faz frente!↵
Lannister cavaleiro (10,9)↵
Sou um heroi Lannister! Vitoria gloriosa!↵
>
```

No exemplo que se segue, o último soldado do reino `Lannister`, o `cavaleiro`, na posição (10, 10) tenta-se deslocar 3 vezes para `sul`, encontrando, sucessivamente, um cavaleiro e um lanceiro do reino `Stark`. O `cavaleiro` derrota o cavaleiro do reino `Stark`, mas depois é derrotado pelo lanceiro. Por ser o último Lannister e os Lannister já não terem castelos, perde o jogo, dando a vitória ao reino `Stark`. Neste cenário, a última jogada é ignorada.

```
Lannister > soldado 10 10 sul sul sul↵
Muhahah, sou um Lannister! Sou invencivel! Nenhum cavaleiro me faz frente!↵
Lannister cavaleiro (9,10)↵
Argh! A dor! Maldito sejas, lanceiro Stark.↵
Lannister cavaleiro morto↵
Sou um heroi Stark! Vitoria gloriosa!↵
>
```

No exemplo que se segue, o espadachim na posição (13, 16) e desloca-se para este, encontrando o castelo do único reino adversário activo `Castle Black`, o qual está livre. Para além disso o reino adversário já não tem soldados.

Lannister > `soldado 13 16 este`↵
O espadachim da ilustre casa de Lannister adquiriu um novo castelo Castle Black para o seu reino.↵
Lannister espadachim(vivo) 14 16↵
Sou um heroi Lannister! Vitoria gloriosa!↵
>

4 Exemplo completo

Nesta secção apresentamos um exemplo de jogo completo. Note que este exemplo não pretende ser exaustivo de tudo o que pode acontecer, mas apenas ilustrar uma sessão de jogo típica.

novo - Novo jogo↵
ajuda - Mostra a ajuda↵
sai - Termina a execucao do programa↵
> novo 10 10 2 3↵
3 castelos:↵
3 2 10 Castle Black↵
6 5 15 Castle Red↵
6 2 7 Castle My↵
2 reinos:↵
TeamBlack Castle Blac↵
TeamRed Castle Red↵
Jogo iniciado, comeca o reino TeamBlack.↵
TeamBlack > mapa↵
10 10↵
3 castelos:↵
Castle Black (TeamBlack)↵
Castle Red (TeamRed)↵
Castle My (sem dono)↵
2 reinos:↵
TeamBlack; TeamRed↵
TeamBlack > recruta cavaleiro Castle Black↵
cavaleiro recrutado no Castle Black do reino TeamBlack por 4 moedas.↵
TeamRed > recruta cavaleiro Castle Red↵
cavaleiro recrutado no Castle Red do reino TeamRed por 4 moedas.↵
TeamBlack > soldado 3 2 norte este este↵
TeamBlack cavaleiro (3,3)↵
TeamBlack cavaleiro (4,3)↵
TeamBlack cavaleiro (5,3)↵
TeamBlack > castelos↵
1 castelos:↵
Castle Red com riqueza 14 na posicao (6,5)↵
TeamRed > soldado 6 5 sul este sul↵
TeamRed cavaleiro (6,4)↵
TeamRed cavaleiro (7,4)↵
TeamRed cavaleiro (7,3)↵
TeamBlack > exercito↵
1 soldados:↵
cavaleiro na posicao (5,3)↵
TeamBlack > recruta lanceiro Castle Black↵
lanceiro recrutado no Castle Black do reino TeamBlack por 2 moedas.↵
TeamRed > exercito↵
1 soldados:↵
cavaleiro na posicao (7,3)↵
TeamRed > soldado 7 3 oeste oeste oeste↵
TeamRed cavaleiro (6,3)↵
Muhahah, sou um TeamRed! Sou invencivel! Nenhum cavaleiro me faz frente!↵
TeamRed cavaleiro (5,3)↵
TeamRed cavaleiro (4,3)↵

TeamBlack > `soldado 3 2 este`↵
TeamBlack lanceiro (4,2)↵
TeamRed > `soldado 4 3 oeste sul sul`↵
TeamRed cavaleiro (3,3)↵
O cavaleiro da ilustre casa de TeamRed adquiriu um novo castelo Castle Black para o seu reino.↵
TeamRed cavaleiro (3,2)↵
TeamRed cavaleiro (3,1)↵
TeamBlack > `castelos`↵
Sem castelos.↵
TeamBlack > `exercito`↵
1 soldados:↵
lanceiro na posicao (4,2)↵
TeamBlack > `soldado 4 2 este`↵
TeamBlack lanceiro (5,2)↵
TeamRed > `castelos`↵
2 castelos:↵
Castle Red com riqueza 20 na posicao (6,5)↵
Castle Black com riqueza 13 na posicao (3,2)↵
TeamRed > `exercito`↵
1 soldados:↵
cavaleiro na posicao (3,1)↵
TeamRed > `recruta espadachim Castle Black`↵
espadachim recrutado no Castle Black do reino TeamRed por 2 moedas.↵
TeamBlack > `exercito`↵
1 soldados:↵
lanceiro na posicao (5,2)↵
TeamBlack > `soldado 5 2 este`↵
O lanceiro da ilustre casa de TeamBlack adquiriu um novo castelo Castle My para o seu reino.↵
TeamBlack lanceiro (6,2)↵
TeamRed > `exercito`↵
2 soldados:↵
cavaleiro na posicao (3,1)↵
espadachim na posicao (3,2)↵
TeamRed > `soldado 3 1 este este este`↵
TeamRed cavaleiro (4,1)↵
TeamRed cavaleiro (5,1)↵
TeamRed cavaleiro (6,1)↵
TeamBlack > `exercito`↵
1 soldados:↵
lanceiro na posicao (6,2)↵
TeamBlack > `soldado 6 2 sul`↵
Muhahah, sou um TeamBlack! Sou invencivel! Nenhum cavaleiro me faz frente!↵
TeamBlack lanceiro (6,1)↵
TeamRed > `exercito`↵
1 soldados:↵
espadachim na posicao (3,2)↵
TeamRed > `castelos`↵
2 castelos:↵
Castle Red com riqueza 24 na posicao (6,5)↵
Castle Black com riqueza 15 na posicao (3,2)↵
TeamRed > `soldado 3 2 sul`↵
TeamRed espadachim (3,1)↵

TeamBlack > `exercito`↵
1 soldados:↵
lanceiro na posicao (6,1)↵
TeamBlack > `soldado 6 1 este`↵
TeamBlack lanceiro (7,1)↵
TeamRed > `soldado 3 1 este`↵
TeamRed espadachim (4,1)↵
TeamBlack > `soldado 7 1 oeste`↵
TeamBlack lanceiro (6,1)↵
TeamRed > `soldado 4 1 este`↵
TeamRed espadachim (5,1)↵
TeamBlack > `soldado 6 1 oeste`↵
Argh! A dor! Maldito sejas, espadachim TeamRed.↵
TeamBlack lanceiro morto↵
TeamRed > `exercito`↵
1 soldados:↵
espadachim na posicao (5,1)↵
TeamRed > `castelos`↵
2 castelos:↵
Castle Red com riqueza 30 na posicao (6,5)↵
Castle Black com riqueza 21 na posicao (3,2)↵
TeamRed > `soldado 5 1 oeste`↵
TeamRed espadachim (4,1)↵
TeamBlack > `exercito`↵
Sem exercito.↵
TeamBlack > `castelos`↵
1 castelos:↵
Castle My com riqueza 27 na posicao (6,2)↵
TeamBlack > `recruta soldado Castle My`↵
Tipo de soldado inexistente.↵
TeamRed > `exercito`↵
1 soldados:↵
espadachim na posicao (4,1)↵
TeamRed > `soldado 4 1 norte`↵
TeamRed espadachim (4,2)↵
TeamBlack > `recruta espadachim Castle My`↵
espadachim recrutado no Castle My do reino TeamBlack por 2 moedas.↵
TeamRed > `soldado 4 2 oeste`↵
TeamRed espadachim (3,2)↵
TeamBlack > `castelos`↵
1 castelos:↵
Castle My com riqueza 29 na posicao (6,2)↵

```
TeamBlack > exercito↵
1 soldados:↵
espadachim na posicao (6,2)↵
TeamBlack > soldado 6 2 sul↵
TeamBlack espadachim (6,1)↵
TeamRed > exercito↵
1 soldados:↵
espadachim na posicao (3,2)↵
TeamRed > castelos↵
2 castelos:↵
Castle Red com riqueza 36 na posicao (6,5)↵
Castle Black com riqueza 27 na posicao (3,2)↵
TeamRed > recruta cavaleiro Castle Red↵
cavaleiro recrutado no Castle Red do reino TeamRed por 4 moedas.↵
TeamBlack > exercito↵
1 soldados:↵
espadachim na posicao (6,1)↵
TeamBlack > soldado 6 1 norte↵
TeamBlack espadachim (6,2)↵
TeamRed > exercito↵
2 soldados:↵
espadachim na posicao (3,2)↵
cavaleiro na posicao (6,5)↵
TeamRed > soldado 6 5 sul sul sul↵
TeamRed cavaleiro (6,4)↵
TeamRed cavaleiro (6,3)↵
Muhahah, sou um TeamRed! Sou invencivel! Nenhum espadachim me faz frente!↵
O cavaleiro da ilustre casa de TeamRed adquiriu um novo castelo Castle My para o seu reino.↵
TeamRed cavaleiro (6,2)↵
Sou um heroi TeamRed! Vitoria gloriosa!↵
> sai↵
Obrigado por jogar. Ate a proxima.↵
↵
```

5 Método de Desenvolvimento do Projeto

Esta secção fornece algumas recomendações sobre uma metodologia de trabalho adequada para um projecto de desenvolvimento de um sistema de software. É muito importante ser metódico em qualquer actividade com alguma duração, particularmente num projecto deste tipo. Acarreta uma sequência de actividades que devem ser feitas de forma disciplinada (ou seja, um processo de desenvolvimento), para se obter um "produto" final de qualidade, minorando a ocorrência de defeitos (bugs) e facilitando a sua detecção quando ocorrem, facilitando a legibilidade do programa, e... ter uma boa nota.

5.1 ETAPA 1 – Compreensão e Esclarecimento do Enunciado e dos Objectivos do Projecto

Leia bem o enunciado, anotando todas as questões que lhe ocorram - de preferência por escrito e em papel. Mantenha um documento escrito com as dúvidas sobre este trabalho e com as respostas que for obtendo. É essencial saber, em cada momento, que dúvidas tem, e quais as respostas às dúvidas que teve, mas já esclareceu.

Como esclarecer dúvidas? Com os docentes da disciplina. Aproveite os horários de dúvidas

e as aulas práticas. Nesta fase eliminará as maiores dúvidas sobre o enunciado e sobre o que se pretende. Claro, algumas dúvidas podem persistir e podem surgir novas questões durante as fases seguintes. Por isso, prepare-se para manter o seu documento de dúvidas até ao fim do prazo de entrega.

5.2 ETAPA 2 – Análise da Estrutura Geral do Programa

Nesta fase, terá como objectivo definir a estrutura global do seu programa. Como bem sabe, um programa em Java consiste num conjunto de classes Java, cujos objectos representam as várias entidades do domínio do problema, e talvez uma ou outra classe do domínio da solução ou implementação.

No caso que temos em mão, além da classe Main, vai existir uma classe do domínio para cada entidade necessária para construir a aplicação. Lembre que aqui falamos de jogos, reinos, castelos, soldados, etc.

Os objectos dessas classes representarão os conceitos em actividade durante a execução do programa. Manterão a informação relativa a cada instância desse conceito, e fornecerão grupos de operações apropriadas. Defina nesta fase quais são as várias classes de que necessita e as suas interfaces no contexto do programa em causa. Certifique-se de que as operações que identificou nas interfaces pertencem mesmo à classe em que as colocou. Identifique também as variáveis e constantes que achar necessárias para representar o estado de cada objecto de cada classe, assim como o seu tipo.

Não se esqueça de pensar bem como representar a informação dentro de cada objecto. Tente explorar o facto de se poderem usar objectos de uma classe como valores de variáveis de outros objectos. Note que nesta fase não terá que programar, mas apenas que pensar e definir, para cada classe, a interface (conjunto de operações), assim como os construtores, as variáveis de instância e constantes que achar necessárias.

Quando tiver dúvidas sobre onde colocar uma determinada operação (em que classe), consulte os docentes da disciplina. Para que tudo fique bem documentado, para cada variável indique o seu tipo e explique a finalidade. Para cada método indique o tipo do seu resultado e o tipo dos seus parâmetros (se existirem). Explique ainda a sua finalidade (para que serve) de forma clara e intuitiva. Indique ainda que métodos são modificadores e que métodos são de consulta.

O resultado desta fase é um "esqueleto" do programa em Java, em que as classes só têm variáveis, métodos (cujo corpo é vazio), e comentários explicativos. É um programa que provavelmente não passa no compilador (que dá erros, pois faltam coisas), mas que é o esqueleto da sua solução. E o esqueleto é o mais importante. Se for sólido e bem pensado, o programa vai-se manter de pé quando concluído, caso contrário, vai sair uma "alforreca".

No final desta fase, mostre o resultado da sua planificação aos docentes da disciplina. Estes poderão dar alguns conselhos, ou transmitir-lhe confiança sobre a adequação da sua proposta. Apenas depois desta fase estar concluída é que deve começar a programar.

5.3 ETAPA 3 – Construção do Interpretador de Comandos

Nesta etapa deve focar-se em construir o interpretador de comandos que servirá para executar o seu programa. Deve replicar da forma mais exata que conseguir o enunciado e os comandos nele descritos. Desta forma, terá um programa com a possibilidade de invocar todas as funcionalidades pedidas.

Assim, nesta fase terá necessariamente de criar uma classe especial – `Main` – que contém o programa principal (método `main`), e diversos métodos auxiliares (por exemplo, gestão do input/output), o que permite ao programa interagir com o utilizador. Pode montar um método

a representar cada comando especificado neste enunciado, mesmo que, numa primeira fase, alguns desses métodos ainda não façam nada. Depois, poderá preencher esses métodos usando as diversas funcionalidades das sua classes lógicas. Trate de testar as funcionalidades/comandos à medida que vai implementando.

5.4 ETAPA 4 – Construção do programa

Nesta fase, vai a programar as classes do domínio da sua aplicação. Desenvolva cada classe separadamente e teste-a bem antes de a integrar na aplicação. Para testar a classe faça um programa principal onde cria objectos e realiza operações nesses objectos, tal como fez nos primeiros exercícios desta unidade curricular.

Para programar as suas classes, comece por fazer um levantamento das constantes úteis, dos métodos da interface (observadores e modificadores/observadores) e das variáveis de instância. Desenvolva depois o algoritmo e finalmente o código de cada método.

Note que não necessita de ter a sua aplicação completa para a poder começar a testar. Os cenários de teste que lhe são fornecidos são construídos, propositadamente, de modo incremental, para que possa ir implementando e testando, aos poucos, as funcionalidades das suas classes. Os cenários serão publicados num documento juntamente com este enunciado na página do moodle.

Na página do moodle será publicado um conjunto de cenários de teste com possíveis interações com o utilizador para estes problemas. Este conjunto de inputs e outputs associados são semelhantes aos que serão testados na plataforma Mooshak.

4.5 Recomendações Gerais

É muito importante tentar ter sempre um programa funcional, que compile sem erros e faça alguma coisa, mesmo que ainda não tudo o que é requerido. Assim terá mais segurança e uma base sólida para partir para a fase seguinte. É a regra da versão estável, que deve ser seguida por qualquer desenvolvedor de software. Tenha sempre uma versão estável do seu sistema. Nunca comece a trabalhar em novas fases, sem que as anteriores estejam robustas, bem testadas.

Para facilitar o desenvolvimento do seu programa, e eliminar erros desnecessários, deverá ter em atenção os seguintes aspectos:

- Não se esqueça de mostrar o resultado da ETAPA 1 a um docente ou monitor. Não comece a programar antes de terminar a ETAPA 1.
- Teste as suas classes método a método. Se for preciso crie pequenos programas principais para testar (os métodos de) cada classe. Aproveite os problemas no Mooshak para testar as suas classes.
- Cada método deve realizar uma (e só uma) tarefa bem definida. Defina métodos auxiliares (privados) sempre que tal clarificar a divisão de tarefas e legibilidade do seu programa.
- Use bons nomes para os identificadores. Siga as regras de estilo adoptadas no standard da disciplina, disponível no moodle.
- Quando definir a classe Main e o respectivo método main, vá adicionando funcionalidades à classe conforme for desenvolvendo as várias funcionalidades do sistema.
- A classe Main deverá ser organizada em vários métodos estáticos e privados, que implementam cada comando do sistema, podendo também haver métodos auxiliares. O estilo da programação é **MESMO** muito importante.

- A classe Main é a única que pode utilizar operações de input/output de ou para o ecrã (println, etc.). Nenhum método de uma classe auxiliar pode conter operações de input/output.
- Todas as variáveis de estado dos objectos (variáveis de instância) TÊM que ser privadas (declaradas private).
- Não se esqueça da regra da "versão estável". No fim de cada fase de desenvolvimento ou alteração ao programa, ele deve compilar e funcionar como se espera nessa fase. Nunca esteja muito tempo sem que o seu programa compile sem erros, ou não faça o que era previsto fazer nessa fase! As versões estáveis são uma rede de segurança. Nunca apague o código de cada versão estável, para poder voltar a ela, se se perder no desenvolvimento ou se começar a ficar com um programa muito confuso.
- Se não percebe como o SEU programa funciona, dificilmente os docentes o vão perceber quando o forem avaliar! Não escreva código que não entende como funciona. Em caso de dúvida, estude, consulte livros, os docentes, os colegas. **MUITO IMPORTANTE**: comente sempre o seu código, incluindo pré-condições.
- Quando, ao compilar, o seu programa der um erro, tente perceber bem porque isso aconteceu, para não repetir o mesmo erro muitas vezes. Leia com atenção as mensagens que lhe são mostradas.

6 A Entrega

O seu programa deve ser entregue no concurso do mooshak para o efeito (Concurso IP201718_TP2), até ao dia 15 de Dezembro de 2017 pelas 17:00 (hora de Lisboa).

Cada grupo de dois alunos deve registar-se neste concurso. O nome do utilizador deve ser os seus **números de alunos** no grupo correspondente ao seu turno. Os número por ordem crescente. Por exemplo, os alunos nº 3435 e nº 3436 do turno p1 devem ter como nome utilizador no mooshak **3435_3436** no grupo **TP1**. Só serão aceites como entregas para avaliação, os programas cujo utilizador no concurso do Mooshak siga estas regras.

Para o trabalho ser avaliado, o grupo de alunos que entregou o trabalho deve ter cumprido o código de ética do DI (ver na página do moodle), e ter, no final do prazo, uma entrega no concurso do mooshak. Pode submeter o seu código mais que uma vez. Será a última submissão que será avaliada.

A avaliação do seu trabalho tem 3 componentes, sendo a nota final calculada como a soma das notas dessas componentes:

- Avaliação da correcção dos resultados produzidos: até 12 valores
  - Submissão ao concurso obtendo a pontuação máxima (100 pontos), com pelo menos duas classes (Main e Jogo) bem definidas terá 12 valores nesta componente. Note que muito provavelmente vai necessitar de mais do que duas classes para fazer este projecto **bem**.
  - Uma submissão com erros em algumas das funcionalidades terá uma classificação correspondente às funcionalidades correctamente implementadas.
- Avaliação da qualidade do código: 8 valores

Por exemplo, um aluno que entregue uma aplicação que receba 90 pontos terá 90% dos 12 pontos, a que se somam entre 0 e 8 pontos pela qualidade do código.

7 Descrição dos ficheiros de teste

Nesta secção descrevemos os ficheiros de teste. Os ficheiros de teste disponibilizados são semelhantes aos que serão usados para testar a sua aplicação no mooshak. Assim, se o seu trabalho passar nos testes disponibilizados, também passa nos do mooshak.

Os testes do Mooshak verificam de forma incremental a implementação dos vários comandos:

Ficheiro de teste `01_in.txt` (5 pontos)
Comandos testados: `novo, ajuda e sai`
Testa os comandos ajuda e novo (criação de um novo jogo sem erros).

Ficheiro de teste `02_in.txt` (5 pontos)
Comandos testados: `novo, ajuda, sai`
Testa os comandos ajuda e novo (criação de um novo jogo com erros na dimensão do mapa, número de reinos e castelos).

Ficheiro de teste `03_in.txt` (5 pontos)
Comandos testados: `novo, ajuda, sai`
Testa os comandos ajuda e novo (criação de um novo jogo com erros nos castelos).

Ficheiro de teste `04_in.txt` (5 pontos)
Comandos testados: `novo, ajuda, sai`
Testa os comandos ajuda e novo (criação de um novo jogo com erros nos reinos).

Ficheiro de teste `05_in.txt` (5 pontos)
Comandos testados: `novo, mapa, ajuda, sai`
Testa o comando mapa após a criação de jogos corretamente.

Ficheiro de teste `06_in.txt` (5 pontos)
Comandos testados: `novo, mapa, ajuda, sai`
Testa o comando mapa após a criação de jogos.

Ficheiro de teste `07_in.txt` (5 pontos)
Comandos testados: `novo, castelos, ajuda, sai`
Testa o comando castelos após a criação de jogos.

Ficheiro de teste `08_in.txt` (5 pontos)
Comandos testados: `novo, exercito, ajuda, sai`
Testa o comando exercito após a criação de jogos.

Ficheiro de teste `09_in.txt` (5 pontos)
Comandos testados: `novo, mapa, castelos, exercito, ajuda, sai`
Testa os comandos mapa, castelos e exercito após a criação de jogos.

Ficheiro de teste `10_in.txt` (5 pontos)
Comandos testados: `novo, reinos, ajuda, sai`
Testa o comando reinos após a criação de jogos.

Ficheiro de teste `11_in.txt` (5 pontos)
Comandos testados: `novo, mapa, castelos, recruta, sai`
Testa o comando recruta em casos simples (sem haver ainda movimentos de soldados).

Ficheiro de teste `12_in.txt` (5 pontos)

Comandos testados: `novo, mapa, castelos, exercito, recruta, sai`

Testa os comandos recruta em casos simples (sem haver ainda movimentos de soldados).

Ficheiro de teste `13_in.txt` (5 pontos)

Comandos testados: `novo, mapa, castelos, exercito, recruta, soldado, sai`

Semelhante ao teste 12, mas fazendo movimentar os soldados de modo a poder ter mais de um soldado nos reinos.

Ficheiro de teste `14_in.txt` (5 pontos)

Comandos testados: `novo, mapa, castelos, exercito, recruta, soldado, sai`

Semelhante ao teste 13, mas detectando erros que envolvem a não movimentação (sem envolver castelos).

Ficheiro de teste `15_in.txt` (5 pontos)

Comandos testados: `novo, mapa, castelos, exercito, recruta, soldado, sai`

Semelhante ao teste 14, mas detectando a captura de castelos.

Ficheiro de teste `16_in.txt` (5 pontos)

Comandos testados: `novo, mapa, castelos, exercito, recruta, soldado, sai`

Semelhante ao teste 15, havendo combates em que o atacante ganha

Ficheiro de teste `17_in.txt` (5 pontos)

Comandos testados: `novo, mapa, castelos, exercito, recruta, soldado, sai`

Semelhante ao teste 15, havendo combates em que o atacante ganha ou perde

Ficheiro de teste `18_in.txt` (5 pontos)

Comandos testados: `novo, mapa, castelos, exercito, recruta, soldado, sai`

Semelhante ao teste 17, mas o jogo pode acabar num combate

Ficheiro de teste `19_in.txt` (5 pontos)

Comandos testados: `novo, mapa, castelos, exercito, recruta, soldado, sai`

Semelhante ao teste 18, mas o jogo pode acabar por conquista de castelo

Ficheiro de teste `20_in.txt` (5 pontos)

Comandos testados: `novo, mapa, castelos, exercito, recruta, soldado, reinos, sai`

Semelhante ao teste 19, mas inclui o comando reinos (programa completo)