Universidade Lusófona de Humanidades e Tecnologias
Fundamentos de Programação
2022/2023
Brena Lima, Bruno Saraiva, Lúcio Studer, Pedro Alves, Rodrigo Correia
v1.0.0

⛺ Jogo das tendas ⛺

Enunciado do projecto prático - Parte 1

Objectivo

Este projecto tem como objectivo o desenvolvimento de um programa (de agora adiante designado por jogo) usando a **linguagem Kotlin**.

Para alcançar estes objetivos, os alunos devem ter em conta tudo o que foram aprendendo em Fundamentos de Programação (FP).

Organização do Trabalho

Este trabalho está dividido em duas partes, sendo ambas de entrega obrigatória.

Este é o enunciado da Primeira Parte.

O enunciado da Segunda Parte será publicado em data posterior no Moodle.

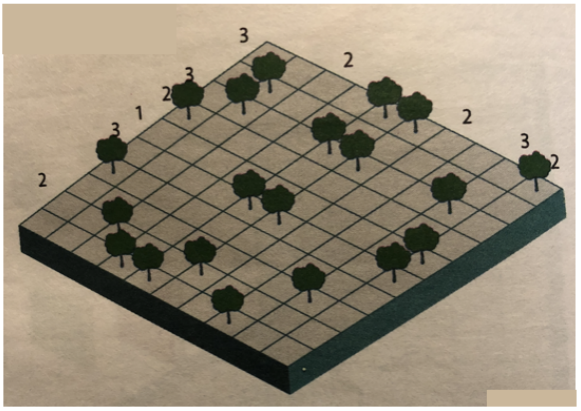
A não entrega de qualquer uma das partes do projecto implica reprovação na componente prática da disciplina.

Tema

A Mensa International é a mais antiga e famosa sociedade de alto QI do mundo. Apenas podem fazer parte desta sociedade pessoas com QI nos 2% de topo mundial, segundo testes de inteligência aprovados.

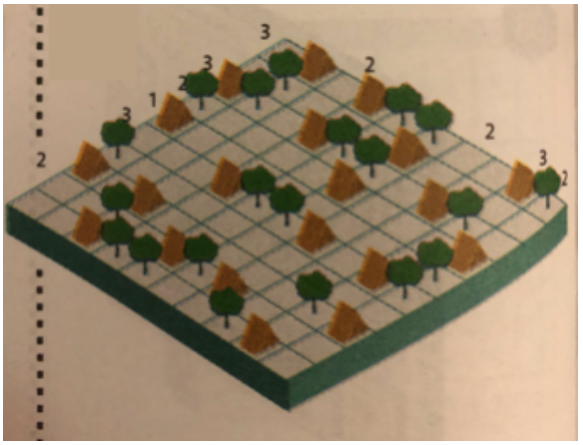
Esta sociedade publica regularmente quebra-cabeças. O tema deste projeto é um desses quebra-cabeças: o jogo das tendas.

Dado um “terreno” quadrado ou rectangular no qual existem árvores dispostas aleatoriamente, similar a esta figura:



Pretende-se que o jogador anexe uma tenda a cada árvore, colocando-a num quadrado (horizontal ou vertical) adjacente à árvore. As tendas não se podem tocar - nem mesmo na diagonal. Os números fora da grelha revelam o número total de tendas nessas linhas ou colunas.

A solução para o mapa da figura anterior seria:



Objectivos da Primeira Parte

Nesta primeira parte do projecto, ainda não vai ser desenvolvido o jogo propriamente dito, até porque ainda não foram ensinadas algumas das técnicas que serão necessárias para o implementar. Pretende-se que os alunos implementem o mecanismo de inicialização e validação do jogo:

- Apresentação no ecrã dos menus do jogo
- Introdução de dados e respectiva validação, nomeadamente:
    - Tamanho do terreno
    - Data de nascimento do jogador
    - Coordenadas onde colocar a tenda
- Apresentação no ecrã do terreno inicial com dados fixos (isto é, as árvores aparecerão sempre na mesma posição)

Notem que a colocação de tendas propriamente dita e restante mecânica do jogo só será implementada na parte 2 do projecto.

Domínio do Problema

A seguir descrevem-se alguns conceitos importantes para a compreensão do enunciado.

Terreno

O jogo decorre num terreno representado por uma grelha MxN, onde M representa o número de linhas e N o número de colunas. Cada elemento da grelha é designado por casa.

As casas do terreno podem conter os seguintes caracteres que passamos a descrever:

△ (código unicode 25B3) - Representa uma árvore

T - Representa uma tenda

Para mais facilmente se poder identificar qual a casa na qual queremos colocar uma tenda (a ser implementado na parte 2), o terreno deve ter uma legenda horizontal e uma legenda vertical.

A legenda horizontal identifica as colunas do terreno, que são letras maiúsculas entre A e Z.

A legenda vertical identifica as linhas do terreno, que são números inteiros sequenciais a começar em 1.

Apresenta-se um exemplo de um terreno 6x5:

	A	B	C	D	E
1					
2					
3					
4					
5					
6					

Apenas serão permitidas algumas configurações de terreno:

- 6x5
- 6x6
- 8x8
- 10x10
- 8x10
- 10x8

Funcionalidades da Primeira Parte

Nesta secção descrevemos em pormenor os objetivos a implementar pelos alunos.

Menus do Jogo

O menu principal deverá ter a seguinte estrutura:

```
Bem vindo ao jogo das tendas

1 - Novo jogo
0 - Sair
```

Notas:

- Nos próximos exemplos, o que for escrito a negrito / *bold*, representa texto que o jogador irá inserir quando estiver a usar o programa.
- Devem respeitar exactamente todas as linhas em branco e os espaços em branco (podem e devem usar *copy & paste* dos exemplos para vos ajudar).

Ao escolher uma opção diferente de **0** ou **1**, o jogo apresenta a mensagem “Opcao invalida” seguido de uma quebra de linha e volta a mostrar o menu.

Ao escolher a opção **0** o jogo termina.

Ao escolher a opção **1** o jogo irá perguntar qual a configuração do terreno.

Configuração do terreno

O programa deverá pedir para o utilizador introduzir o número de linhas, seguido do número de colunas.

Exemplo:

```
Bem vindo ao jogo das tendas

1 - Novo jogo
0 - Sair

1
Quantas linhas?
8
Quantas colunas?
8
```

Quer o número de linhas quer o número de colunas têm que ser números inteiros superiores a zero. Caso isso não aconteça, deve ser mostrado uma mensagem “Resposta invalida” e voltar a perguntar. Isto deve acontecer até ser introduzida uma resposta válida.

Exemplos:

```
Bem vindo ao jogo das tendas

1 - Novo jogo
0 - Sair

1
Quantas linhas?
ola
Resposta invalida
Quantas linhas?
-5
Resposta invalida
Quantas linhas?
```

```
Bem vindo ao jogo das tendas

1 - Novo jogo
0 - Sair

1
Quantas linhas?
10
Quantas colunas?
banana
Resposta invalida
Quantas colunas?
```

Após o número de linhas e colunas ser corretamente introduzido, deverá ser validado se corresponde a uma configuração de terreno válida (ver secção Terreno). Caso não seja, deve ser mostrada a mensagem “Terreno invalido” e voltar ao menu principal.

Exemplo:

```
Bem vindo ao jogo das tendas

1 - Novo jogo
0 - Sair

1
Quantas linhas?
```

```
7
Quantas colunas?
7
Terreno invalido

Bem vindo ao jogo das tendas

1 - Novo jogo
0 - Sair
```

Validação da idade

Caso a configuração seja válida, há ainda um passo adicional antes de mostrar o terreno. Caso o terreno tenha a configuração 10x10 (e apenas nesse caso), deverá pedir ao jogador para introduzir a data de nascimento, no formato dd-mm-yyyy[1]. A ideia é que, para terrenos mais difíceis (como é o caso do 10x10), o jogador tem que ter mais de 18 anos[2]. Caso não tenha mais de 18 anos, deverá mostrar uma mensagem “Menor de idade nao pode jogar” e voltar ao menu principal.

Exemplo:

```
Bem vindo ao jogo das tendas

1 - Novo jogo
0 - Sair

1
Quantas linhas?
10
Quantas colunas?
10
Qual a sua data de nascimento? (dd-mm-yyyy)
01-01-2015
Menor de idade nao pode jogar

Bem vindo ao jogo das tendas

1 - Novo jogo
0 - Sair
```

A data terá que ser validada. Caso o utilizador introduza uma data inválida, deverá voltar a perguntar a data.

[1] Significa que o dia e mês serão sempre representados com 2 dígitos. Ou seja, 01-01-2000 e não 1-1-2000.
[2] Para simplificar a implementação, podem assumir que qualquer data de nascimento igual ou superior a 01-11-2004 é menor de 18 anos.

Exemplo (apenas da parte da data):

```
...
Qual a sua data de nascimento? (dd-mm-yyyy)
30
Data invalida
Qual a sua data de nascimento? (dd-mm-yyyy)
40-12-2000
Data invalida
Qual a sua data de nascimento? (dd-mm-yyyy)
```

O ano deverá ser superior a 1900 e inferior ou igual a 2022. Não esquecer que os meses têm um número de dias diferentes e que, no caso de Fevereiro, é preciso ter em conta o ano (lembrem-se do exercício “quantos dias tem o mês?” que fizemos nas aulas).

Finalmente, se o jogador tem mais de 18 anos (ou se a configuração não obriga a perguntar a idade), deve ser mostrado o terreno.

Visualização do terreno

O terreno deve refletir o tamanho escolhido previamente. Após o terreno, deve ser perguntado ao jogador quais as coordenadas onde quer colocar a tenda.

Notem que nesta primeira parte, o terreno vai estar preenchido de forma fixa, com árvores na última coluna de cada linha, como se pode ver no exemplo:

```
Bem vindo ao jogo das tendas

1 - Novo jogo
0 - Sair

1
Quantas linhas?
6
Quantas colunas?
6

   | A | B | C | D | E | F
 1 |   |   |   |   |   | △
 2 |   |   |   |   |   | △
 3 |   |   |   |   |   | △
 4 |   |   |   |   |   | △
 5 |   |   |   |   |   | △
```

```
 6 |   |   |   |   |   | △
Coordenadas da tenda? (ex: 1,B)
```

Notas:

- As casas no terreno têm sempre o comprimento correspondente a 3 espaços.
- Devem contabilizar uma linha extra em cima do terreno.
- Devem contabilizar um espaço extra no início de cada linha, para garantir que quando chega à linha 10 não fica desformatado.
- Devem usar o caracter '|' para servir de separador das colunas

Exemplo do terreno com 10 linhas:

```
   | A | B | C | D | E | F | G | H | I | J
 1 |   |   |   |   |   |   |   |   |   | △
 2 |   |   |   |   |   |   |   |   |   | △
 3 |   |   |   |   |   |   |   |   |   | △
 4 |   |   |   |   |   |   |   |   |   | △
 5 |   |   |   |   |   |   |   |   |   | △
 6 |   |   |   |   |   |   |   |   |   | △
 7 |   |   |   |   |   |   |   |   |   | △
 8 |   |   |   |   |   |   |   |   |   | △
 9 |   |   |   |   |   |   |   |   |   | △
10 |   |   |   |   |   |   |   |   |   | △
```

Introdução de coordenadas

O jogador deve agora introduzir as coordenadas onde quer colocar a tenda. Nesta primeira parte, deverão apenas validar que o jogador introduziu coordenadas válidas. Neste caso, as coordenadas são válidas se:

- Tiverem o formato <LINHA>,<COLUNA> em que <LINHA> é um inteiro e <COLUNA> é uma letra maiúscula
- A linha fôr igual ou superior a 1 e igual ou inferior ao número de linhas do terreno
- A coluna fôr igual ou superior a A e igual ou inferior à letra correspondente ao número de colunas do terreno

Caso as coordenadas não sejam válidas, deve mostrar a mensagem "Coordenadas invalidas" e voltar a pedir as coordenadas.

Caso sejam válidas, deve voltar ao menu principal. Para já é só isto! Na parte 2 iremos processar estas coordenadas e mudar o terreno de forma a incluir a tenda nessas coordenadas.

Exemplo (apenas da parte das coordenadas):

```
...
Coordenadas da tenda? (ex: 1,B)
4
Coordenadas invalidas
Coordenadas da tenda? (ex: 1,B)
1,C

Bem vindo ao jogo das tendas

1 - Novo jogo
0 - Sair
```

Notas de Implementação

Conversão de caracteres para números

Para implementarem o processamento das colunas, vão precisar de converter letras para números.

Para isso, podem obter o código ascii de qualquer caracter através da seguinte instrução:

```
val ch = 'A'
val codigoAscii: Int = ch.code
```

Notem que "A" é diferente de 'A'. Esta técnica apenas se aplica a Char. Caso tenham as letras em Strings, têm que convertê-las primeiro para Char.

Notem também que o código ascii do caracter 'A' não é 1, por isso poderão ter que fazer mais algum processamento.

Funções de Implementação Obrigatória

Seguindo as boas práticas da programação, o projecto deverá fazer uso intensivo de funções de forma a facilitar a legibilidade e compreensão do mesmo.

Para facilitar esse processo, descrevem-se de seguida um conjunto de funções obrigatórias que deverão implementar e usar para atingir os objetivos enunciados na secção anterior.

Nota importante: Estas funções não devem escrever nada no ecrã (ou seja, não devem usar as funções `print` / `println`).

Assinatura	**Comportamento**
`fun criaMenu(): String`	Retorna uma string das opções do menu (“Novo jogo” e “Sair”) assim como o “Bem vindo ao jogo das tendas”, tal como é apresentado na secção “Menus do Jogo”.
`fun validaTamanhoMapa(numLinhas: Int, numColunas: Int): Boolean`	Retorna true caso a configuração apresentada seja uma configuração válida, de acordo com as regras apresentadas na secção “Terreno”.
`fun validaDataNascimento(data: String?) : String?`	Retorna null caso a String passada como parâmetro contenha uma data válida. Caso a data seja inválida, deve retornar uma de 2 Strings: ● “Data invalida” ● “Menor de idade nao pode jogar” Verifiquem as validações da data na secção “Validação da idade”
`fun criaLegendaHorizontal(numColunas : Int): String`	Devolve a string da legenda horizontal (as letras de A a Z que estão acima do terreno apresentado), sem espaços à volta. Exemplo: `“A | B | C”`
`fun criaTerreno(numLinhas: Int, numColunas: Int, mostraLegendaHorizontal: Boolean, mostraLegendaVertical: Boolean): String`	Retorna uma string com a representação do terreno, tendo em conta a configuração indicada. O parâmetro `mostraLegendaHorizontal` indica se deve ser mostrada a linha com as letras A | B | … em cima do terreno. O parâmetro `mostraLegendaVertical` indica se deve ser mostrada a coluna com os números: 1 2 3 do lado esquerdo do terreno. Esta função deve chamar a função criaLegendaHorizontal(...) Importante: A função deve estar preparada para ser executada apenas com 2 ou 3 parâmetros, usando valores por omissão. Os valores por omissão para o 3º e 4º parâmetros são true e true respetivamente.
`fun processaCoordenadas(coordenadasStr: String?, numLines: Int, numColumns: Int): Boolean`	Verifica se as coordenadas introduzidas pelo jogador são válidas, tendo em conta a configuração do terreno.

Podem e devem implementar funções adicionais se isso ajudar a organizar o vosso código. Em particular, recomendamos que criem funções adicionais para processar cada um dos inputs. Ou seja, que validam as respostas a uma certa pergunta específica num ciclo que só termina quando a resposta é válida (e retorna essa resposta).

Funções com demasiadas linhas de código (incluindo a `main`) levarão a penalizações na componente de qualidade de código.

Entrega e Avaliação

Artefactos a Entregar

A entrega deve ser feita através do Drop Project usando um ficheiro comprimido (formato .zip) dentro do qual se encontrem:

- Uma pasta com o nome "src" com todo o código necessário para compilar e executar o projecto.
- Um ficheiro de texto (chamado AUTHORS.txt) contendo os nomes e números de aluno).

Nota importante: o ficheiro .zip não deve incluir outras pastas do projecto automaticamente criadas pelo Intellij: `.idea, lib, out`, etc.

Formatos de Entrega

O ficheiro .zip deve seguir a estrutura que se indica de seguida:

```
AUTHORS.txt (contém linhas NUMERO_ALUNO;NOME_ALUNO, uma por aluno do grupo)

+ src
|--- Main.kt
|--- ...   (outros ficheiros kotlin do projecto)
```

Restrições Técnicas

O projecto deverá utilizar apenas as instruções ensinadas nas aulas até à aula 9. **Nomeadamente não deverão ser utilizados ciclos for, arrays, indexOf(), contains(), split(), listas, maps, Pairs, etc. Também não devem usar classes relacionadas com processamento de datas como LocalDate ou SimpleDateFormat.** Não deverão criar classes novas. Projectos que usem estas instruções vão ter fortes penalizações. Na dúvida, deverão contactar o vosso professor das aulas práticas.

A versão do Kotlin ensinada nas aulas e que é oficialmente suportada pelo Drop Project é a versão 1.7.X.

Como Entregar

O projecto será entregue via Drop Project (DP), através do link:

https://deisi.ulusofona.pt/drop-project/upload/fp-projeto-22-23-p1

Não serão aceitas entregas por outra via.

Os alunos são incentivados a testar o projecto no DP à medida que o vão implementando. Podem e devem fazer quantas submissões acharem necessárias. Para efeitos de avaliação será considerada a melhor submissão feita dentro do prazo.

De notar que os alunos, antes de enviarem para o DP, devem testar no seu próprio computador.

Prazos de Entrega

A data limite de entrega é o dia **5 de Dezembro de 2022**, pelas **8h00** (hora de Lisboa, Portugal).

Os alunos que entreguem depois do prazo, ainda durante o dia 5 de Dezembro de 2022, até às 23h30, **terão uma penalização de 5 valores** na nota final desta parte do projecto.

Não serão aceites entregas após dia 5 de Dezembro de 2022 às 23h30. Nesse caso os alunos terão nota zero no projecto, **reprovando na componente prática da disciplina (1ª época)**.

Avaliação

A avaliação do projecto será feita considerando as entregas feitas pelos alunos em ambas as partes. A nota será divulgada no final de cada entrega.

Após a entrega final, será atribuída ao projecto uma nota final quantitativa, que será calculada considerando a seguinte fórmula:

```
0.3 * NotaParte1 + 0.7 * NotaParte2
```

Cotações da Primeira Parte

A avaliação do projecto será feita através dos seguintes tópicos:

Tópico	Descrição	Pontuação (0..20)
Qualidade de código	O código cumpre as boas práticas de programação ensinadas nas aulas (nomes apropriados para as variáveis e funções em camelCase, utilização do val e do var, código bem indentado, funções que não sejam demasiado grandes, evitar usar o !!, etc.).	3
Testes automáticos	Será aplicada uma bateria de testes automáticos cujo relatório poderá ser consultado após cada submissão. Quantos mais testes passarem, melhor nota terão nesta componente.	14
Avaliação manual da aplicação	Os professores farão testes adicionais assim como inspeção de código para avaliar esta componente	3

Cópias

Trabalhos que sejam identificados como cópias serão anulados e os alunos que os submetam terão nota zero em ambas as partes do projecto (quer tenham copiado, quer tenham deixado copiar). **Para evitar situações deste género, recomendamos aos alunos que nunca partilhem ou mostrem o código do seu projecto a pessoas fora do grupo de trabalho.**

A decisão sobre se um trabalho é uma cópia cabe exclusivamente aos docentes da unidade curricular.

Metodologia Recomendada

Os testes do Drop Project fazem 2 tipos de validação: testes de função e testes interativos.

Os testes de função validam que as funções obrigatórias estão bem implementadas, executando diretamente essas funções com parâmetros diferentes e verificando que o retorno é o esperado.

Os testes interativos validam um fluxo completo do jogo, desde que mostra o menu, escolhe uma opção, vai respondendo às questões colocadas até que termina mostrando o terreno. Ou seja, testam a execução do main()

A nossa sugestão é que se concentrem primeiro em passar os testes de função (identificados no relatório do Drop Project com TestTeacherFunctions) e só depois se preocupem com os testes interativos (identificados no relatório do Drop Project com TestTeacherInteractive).

A metodologia a seguir deve ser então:

1 - Comecem por implementar todas as funções obrigatórias de forma "*dummy*", isto é, com uma implementação mínima que compile (ex: a função retornar apenas uma String vazia). Submetam no Drop Project e verifiquem se não há erros de compilação. Deixem a função `main()` vazia.

2 - Vão implementando as funções obrigatórias (agora de forma correta), uma a uma. Para cada função que implementarem, comecem por testá-la no vosso computador. Para isso, devem chamar a vossa função na `main()` e imprimir o resultado, para ver se está correta. Por exemplo, para testarem a função `validaDataNascimento()`, podem fazer algo deste género:

```
fun main() {
  println(validaDataNascimento("3")) // deve escrever "Data invalida"
  println(validaDataNascimento("03-04-1980"))  // deve escrever null
}
```

3 - Após testarem no vosso computador, submetam ao Drop Project, para validar que realmente a função está a funcionar bem (os testes relacionados com essa função já deverão passar). Vão fazendo isso função a função, até passarem todos os testes de função.

4 - Neste ponto, deverá estar a faltar apenas a implementação do jogo propriamente dito, em que o jogador vai introduzindo informações e o programa vai reagindo. Essa implementação deve ser feita na `main()`, usando as funções que desenvolveram previamente. Pensem que já têm algumas peças aparentemente desconexas de um puzzle, que vão agora montar.

5 - Com o `main()` implementado, devem agora concentrar-se em passar os testes interativos. Mas devem sempre testar primeiro no vosso computador. Por exemplo, repitam no vosso programa os exemplos apresentados neste enunciado e verifiquem se o output produzido coincide exatamente com os exemplos. Basta um espaço a mais para falharem no teste.

Outras Informações Relevantes

- Os projetos devem ser realizados em grupos de 2 alunos, preferencialmente da mesma turma prática. Excepcionalmente poderão ser aceites trabalhos individuais, desde que seja pedida autorização prévia ao Professor das aulas práticas respetivo e desde que exista uma justificação razoável.

- O grupo é formado automaticamente pelo Drop Project quando fazem a primeira submissão (através do ficheiro AUTHORS.txt). Submissões individuais que não tenham sido previamente autorizadas por um professor serão eliminadas do Drop Project.

- Existirá uma defesa presencial e individual do projecto, realizada após a entrega da 2ª parte. Durante esta defesa individual, será pedido ao aluno que faça alterações ao código para dar resposta a alterações aos requisitos. Da discussão presencial de cada aluno, resultará uma nota de 0 a 100%, que será aplicada à nota do projecto (primeira e segunda parte).

- Na segunda parte do projecto devem-se manter os grupos definidos para a primeira parte. Não será permitida a entrada de novos membros nos grupos.

- O ficheiro .zip entregue deve seguir escrupulosamente as regras de estrutura indicadas neste enunciado. Ficheiros que não respeitem essa estrutura terão nota zero.

- Trabalhos cujo código não compile e/ou não corra não serão avaliados e terão automaticamente nota zero.

- Grupos que não entreguem a primeira parte ou tenham nota zero na mesma (seja por cópia, não compilação ou outra das situações indicadas no enunciado), não podem entregar a segunda parte do projecto, e reprovam automaticamente na componente prática da disciplina (de primeira época).

- É possível que sejam feitas alterações a este enunciado, durante o tempo de desenvolvimento do projecto. Por esta razão, os alunos devem estar atentos ao Moodle de FP.

- Todas as dúvidas devem ser colocadas no discord.

- Eventuais situações omissas e/ou imprevistas serão analisadas caso a caso pelos docentes da cadeira.