|  |
| --- |
|  |
| Programação Orientada a Objetos |
|  |
| Trabalho Prático |

**ISEC**

2018/2019

Criado por: Guilherme Curado Lousada Pais da Silva -21260315 | Gabriel Marco Freire Pinheiro – 21260736 P1

Programação Orientada a Objetos

Trabalho Prático

# Introdução

Este trabalho visa desenvolver um jogo sobre expedições marítimas, com pescadores, piratas e batalhas navais. Apenas há um jogador, o qual possui diversos navios que conduz a lugares espalhados pelo mundo, realizando pescarias, vendendo peixe, transportando mercadorias, encontrando tesouros e ainda combatendo navios inimigos. Existem diversas adversidades, tais como tempestades e ataques de navios inimigos.

Este trabalho foi desenvolvido em ambiente Unix com recurso à biblioteca *ncurses* para criar uma interface gráfica mais apelativa para o utilizador.

# Organização do código apresentado

## Quais foram as classes consideradas na primeira versão da aplicação que foi testada?

Nesta primeira versão foram desenvolvidas as seguintes classes:

* Interface
* Simulação
* Unidade
* Barco
* Porto

## Quais os conceitos/classe que identificou ao ler o enunciado?

Imediatamente após a leitura do enunciado percebemos que vamos ter que desenvolver uma classe que contenha todos os dados, que neste caso será a classe Simulação, e, consequentemente, serão também desenvolvidas diferentes classes que contenham dados específicos. Neste contexto identificámos que vão existir diversas unidades a navegar no mapa que vão ter alguns parâmetros em comum, como a posição, o aspeto visual e o indicador que diz se são uma unidade amiga ou inimiga, e por isso desenvolvemos a classe Unidade que serve como base para outras duas classes também desenvolvidas: Barco e Porto.

Finalmente percebe-se que tudo isto tem que ser envolto numa interface, e por isso mesmo vai ser criada uma classe Interface que possui toda a parte de interação direta com o utilizador.

## Relativamente a duas das principais classes da aplicação, identifique em que classes ou partes do programa são criados, armazenados e destruídos os seus objetos

As duas principais classes do programa são sem sombra de dúvida a Interface e a Simulação.

Quanto à Interface é criada uma instância da mesma na função *main* do programa, e consequentemente será destruída no final do decorrer do programa.

Já a Simulação vai ver uma instância criada dentro da Interface, ou seja, a Interface possui uma Simulação, e isto é feito para que a Interface tenha acesso a todos os dados que necessita da simulação.

## Indique um exemplo de uma responsabilidade atribuída a uma classe que esteja de acordo com a orientação dada acerca de Encapsulamento

Um bom exemplo disto será a classe Simulação, que possui todos os seus dados no campo privado, e, consequentemente, precisa de uma interface de funções de acesso para os dados que precisarão de ser acedidos exteriormente à própria classe (*Exemplo: A Interface precisa de saber o tamanho do mapa).*

## De entre as classes que fez, escolha duas e justifique por que considera que são classes com objetivo focado, coeso e sem dispersão

Duas classes que não têm qualquer tipo de dispersão quanto ao seu objetivo serão por exemplo a classe Interface e a classe Barco.

Quanto à Interface, esta apenas trata de receber e mostrar dados ao utilizador, pois toda a parte da lógica será tratada pela simulação em si. As funcionalidades da interface passam por ler os comandos introduzidos pelo utilizador e verificar a sua validade, e depois pedir à simulação para executar alguma das suas diversas funções para finalmente mostrar as mudanças efetuadas.

A classe Barco apenas serve de banco de todos os dados que todos os barcos terão, como por exemplo o tipo e o número de soldados, pois toda a lógica que é preciso fazer com os seus dados é feita pela simulação.

## Relativamente à aplicação entregue, quais as classes que considera com responsabilidades de interface com o utilizador e quais as que representam a lógica?

Como acabou por ficar explícito na resposta à pergunta anterior, a classe que serve de Interface com o utilizador será, precisamente, a classe Interface, e a classe que trata de todos os aspetos lógicos será a classe simulação, que executa todas as funções que sejam necessárias ao bom correr do programa.

## Identifique o primeiro objeto para além da camada de interação com o utilizador que recebe e coordena uma funcionalidade de natureza lógica

O primeiro objeto a tratar deste tipo de operações será a Simulação, que logo no arranque do programa e após a inserção do primeiro comando trata de validar o mapa e os dados lidos.

## A classe que representa a envolvente de toda a lógica executa em pormenor muitas funcionalidades, ou delega noutras classes? Indique um exemplo em que esta classe delega uma funcionalidade noutra classe

A classe Simulação será a responsável por toda a lógica, logo não existem delegações de funções noutras classes.

# Principais classes da aplicação

**Classe:** Interface

**Responsabilidades:**

* Leitura e verificação de comandos;
* Visualização gráfica do mapa e constituintes do mesmo;
* Visualização de um *log* com resultado dos comandos inseridos.

**Colaborações:** Possui uma instância da classe Simulação para poder executar as suas funções

**Classe:** Simulação

**Responsabilidades:**

* Consulta do mapa completo (mapa com unidades);
* Toda a lógica da simulação;
* Banco de dados para todos os dados da simulação.
* Possui uma interface para aceder a alguns dos seus dados

**Colaborações:** Possui um vetor de unidades onde guarda todas as unidades da simulação

**Classe:** Unidade

**Responsabilidades:**

* Movimentação para uma dada posição
* Possui uma interface para aceder a todos os seus dados

**Colaborações:** É herdada pelas classes Barco e Porto

**Classe:** Barco

**Responsabilidades:**

* Por enquanto apenas serve como banco de dados
* Possui uma interface para aceder a todos os seus dados

**Colaborações:** Herda a classe Unidade e todos os seus dados e funções

**Classe:** Porto

**Responsabilidades:**

* Por enquanto serve apenas como banco de dados
* Possui uma interface para aceder a todos os seus dados

**Colaborações:** Possui uma instância da classe Simulação para poder executar as suas funções.

# Funcionalidades implementadas

## Configuração/leitura do mapa

A parte de configuração e leitura do mapa está desenvolvida na plenitude. O programa ao receber o comando de configuração acompanhado do ficheiro de texto onde se encontram os dados segue para a validação desses mesmos dados, e então foram definidos os seguintes tipos de erros:

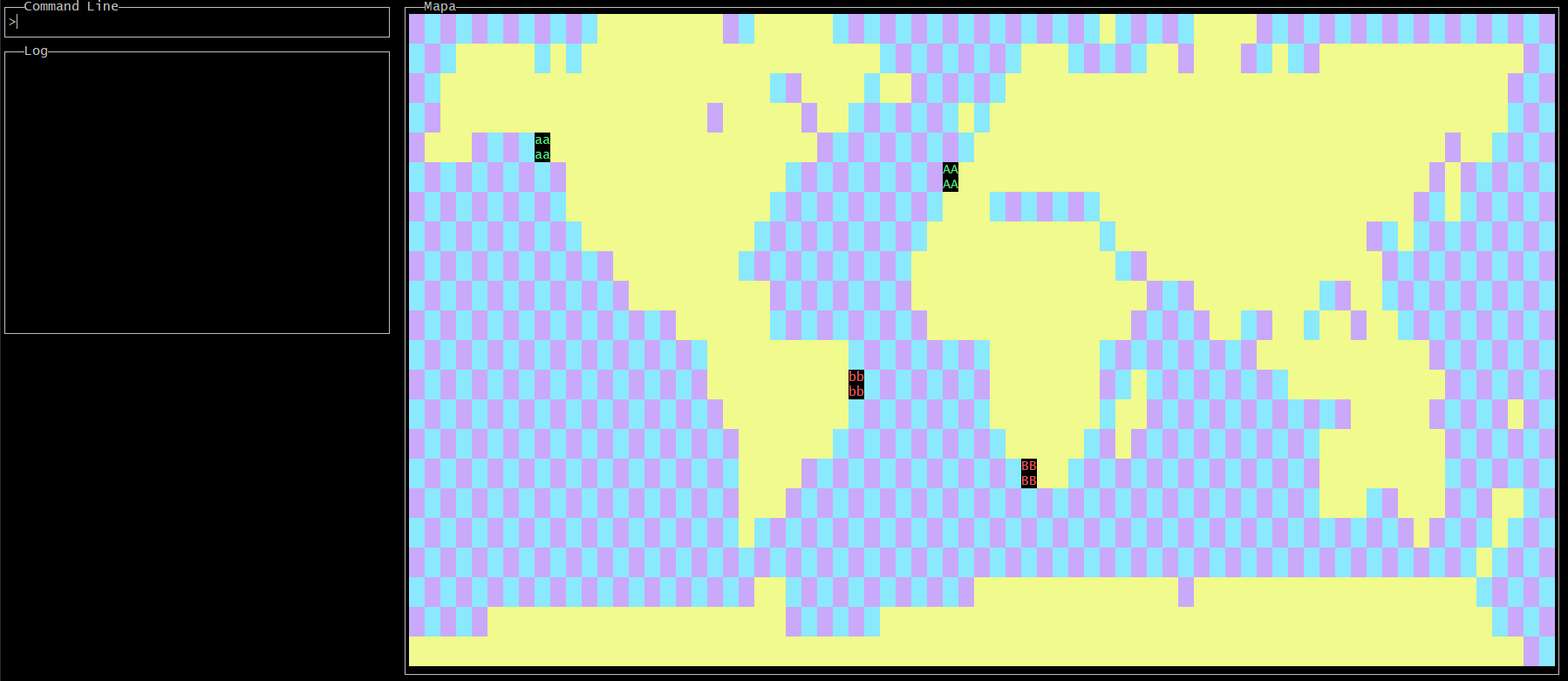
* Erro de abertura
* Erro de formatação
* Erro de fecho
* Erro de valores
* Erro incompleto (quando faltam dados)

Estes erros serão detetados caso alguma coisa não corra como é esperado, mas, caso contrário, todos os dados serão guardados na simulação e a interface avança para o estado seguinte: o início da simulação.

## Visualização do mapa

Esta parte também está completamente feita, mas desta vez apenas para a primeira meta, pois terão de ser feitas alterações para a entrega final.

Após a configuração da simulação o mapa é mostrado durante todo o decorrer da aplicação com cores diferentes para cada elemento presente no mesmo, como podemos ver na imagem seguinte



## Interpretação de comandos

Todos os comandos pedidos no enunciado são lidos e interpretados, sendo que aqueles que não são relevantes para esta meta ainda não possuem qualquer tipo de funcionalidade.

## Compra de navios

Podem ser comprados navios de todos os tipos, menos do tipo especial, pois ainda não decidimos quais vão ser as suas características. Quanto à compra em si apenas será possível se o número de moedas for superior ou igual ao preço do mesmo, caso contrário aparecerá uma mensagem de erro. Ao serem comprados não serão logo visíveis pois estarão na mesma posição do porto.

## Visualização da movimentação dos navios

Todas as movimentações dos navios são visíveis no mapa já descrito anteriormente. A única coisa que ainda vai ser melhorada é o fundo da célula em que os navios aparecem, pois o azul da água não muda independentemente da sua posição, e isso terá de ser resolvido.