Trabalho Prático de Programação Orientada a Objetos

Meta 1

2020/2021

Constituição do grupo

Diogo Florêncio Aires de Vilhena 2018020207 P2
Inês Sofia dos Santos Teixeira 2016016178 P2

Índice

1. Organização do código apresentado

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

Na primeira versão foram consideradas as seguintes classes responsáveis pelas seguintes funcionalidades:

- **Interface**
  - Armazenar e gerir toda a informação necessária à interação com o utilizador;
- **Logica**
  - Armazenar e gerir toda a informação e lógica necessária ao funcionamento do programa;
- **Mundo**
  - Armazenar toda a informação do Mundo, contendo, entre outras coisas, os Territórios existentes no mundo e permitir o acesso a essa mesma informação;
- **Imperio_Jogador**
  - Armazenar toda a informação do Império do Jogador, contendo, entre outras coisas, a Força Militar, Cofre, Armazém e Territórios Conquistados e permitir o acesso a essa mesma informação;
- **Territorio**
  - Armazenar a informação acerca de um território e permitir o acesso à mesma;
- **Armazem**
  - Armazenar as informações relacionadas com o Armazém de produtos do Império e permitir o acesso às mesmas;
- **Cofre**
  - Armazenar as informações relacionadas com o Cofre de ouro do Império e permitir o acesso às mesmas;
- **Forca_Militar**
  - Armazenar as informações relacionadas com a Força Militar e permitir o acesso às mesmas.

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

Durante a leitura do enunciado foram identificas as seguintes classes/conceitos:

- Interface;
- Mundo;
- Territorio;
- Imperio_Jogador;
- Cofre;
- Armazem;
- Forca_Militar.

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

Interface:

- Os objetos desta classe são criados, armazenados e destruídos na Main.

Lógica:

- Os objetos desta classe são criados e destruídos na Main, e armazenados na interface.

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

A responsabilidade "***adicionar um novo Território ao Império***" está atribuída à classe **Logica** porque é onde se encontram tanto o objeto **Mundo**, onde estão guardados os Territórios existentes, como o objeto **Imperio_Jogador** onde estão guardados os Territórios Conquistados.

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

Classe Cofre:

- Tem dados e responsabilidades relativos apenas ao Cofre do Império

Classe Armazém:

- Tem dados e responsabilidades relativos apenas ao Armazém do Império

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

A classe que consideramos com responsabilidade de interface com o utilizador é a classe **Interface**.
A classe que consideramos com responsabilidade de lógica é a classe **Logica**.

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

Os comandos/ordens vindos da camada de interação com o utilizador são recebidos e processados por um objeto da classe Logica.

1.8. 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 **Logica** delega funcionalidades noutras classes, por exemplo a Classe **Mundo** e a Classe **Imperio_Jogador**.
Um exemplo disso é que para verificar se um território já está conquistado delega essa verificação chamando essa função de verificação que pertence à Classe **Imperio_Jogador**.

1.9. Apresente as principais classes da aplicação através da seguinte informação:

Classe: Mundo

Responsabilidades:

- Criar e guardar os Territórios que constituem o mundo;
- Guardar os tipos de Territórios válidos;
- Obter o número de Territórios existentes;
- Obter o nome de um território segundo a posição;
- Obter a resistência de um território segundo a posição;
- Obter um Território segundo o nome;
- Gerar um nome Concatenado para o novo Território;
- Adicionar um Território;
- Verificar se o tipo de Território inserido é válido;
- Verificar se existem Territórios no Mundo;
- Verificar se um determinado Território existe.

Colaborações:

- Territorio.

Classe: Imperio_Jogador

Responsabilidades:

- Guardar os territórios conquistados;
- Criar e guardar o objeto da Força Militar;
- Criar e guardar o objeto do Cofre;
- Criar e guardar o objeto do Armazém;
- Guardar o fator sorte;
- Guardar a produção de ouro nesse turno;
- Guardar a produção de produtos nesse turno;
- Obter fator sorte;
- Obter a força militar;
- Obter o limite máximo da força militar;
- Obter o limite atual da força militar;
- Obter o número de produtos;
- Obter o limite atual do armazém;
- Obter o limite máximo do armazém;
- Obter a quantidade de ouro;
- Obter o limite atual do cofre;
- Obter o limite máximo do cofre;
- Obter o número de territórios conquistados;
- Obter o nome de um território segundo a posição;
- Obter a resistência de um território segundo a posição;
- Obter o ouro produzido por ronda de um território segundo a posição;
- Obter o número de produtos produzido por ronda de um território segundo a posição;
- Obter os pontos de vitória de um território segundo a posição;
- Obter a produção total de ouro nessa ronda;
- Obter a produção total de produtos nessa ronda;
- Gerar fator sorte;
- Incrementar a força militar;
- Incrementar o limite atual da força militar;
- Incrementar o número de produtos;
- Incrementar o limite atual do armazém;
- Incrementar o ouro;
- Incrementar o limite atual do cofre;
- Decrementar a força militar;
- Decrementar o número de produtos;
- Decrementar o ouro;
- Adicionar território conquistado;
- Verifica se o vetor de territórios conquistados está vazio;
- Verifica se o território está conquistado;

- Atualizar a produção de ouro;
- Atualizar a produção de Produtos;
- Encontrar um território segundo o nome.

Colaborações:

- Forca_Militar;
- Cofre;
- Armazem;
- Territorio.

Classe: Territorio

Responsabilidades:

- Guardar o nome;
- Guardar a resistência;
- Guardar os pontos de vitória;
- Guardar a produção de ouro por ronda;
- Guardar a produção de produtos por ronda;
- Obter o nome
- Obter a resistência;
- Obter os pontos de vitória;
- Obter a produção de ouro por ronda;
- Obter a produção de produtos por ronda.

Colaborações:

Classe: Logica

Responsabilidades:

- Guardar o número máximo de turnos;
- Guardar o turno atual;
- Guardar o ano atual;
- Guardar a pontuação final;
- Criar e guardar o Mundo;
- Criar e guardar Império do Jogador;
- Obter o turno atual;
- Obter o ano atual;
- Obter a pontuação final;
- Obter o Império do Jogador;
- Obter o Mundo;
- Adicionar um novo Território ao Império;
- Incrementar o turno;
- Incrementar o ano;
- Criar N territórios;
- Dividir o comando em tokens;
- Carregar o ficheiro;
- Preparar o jogo;

- Verificar se o território está conquistado;
- Verificar se todos os territórios já estão conquistados;
- Calcular a pontuação final;
- Conquistar território.

Colaborações:

- Imperio_Jogador;
- Mundo.

Classe: Interface

Responsabilidades:

- Guardar o objeto Logica;
- Lançar e mostrar o Menu Inicial;
- Lançar e mostrar o Menu do Jogo;
- Lançar e mostrar o Menu da Primeira Fase;
- Lançar e mostrar o Menu da Segunda Fase;
- Listar as informações iniciais;
- Listar as informações no jogo;
- Listar as informações de um só território;
- Obter os comandos/ações do utilizador.

Colaborações:

- Logica.

2. Funcionalidades Implementadas

Componente do Trabalho	Realizado	Realizado Parcialmente	Não Realizado
Comando “cria”	X		
Comando “carrega”	X		
Comando “conquista”	X		
Comando “lista”	X		
Projeto organizado em .h e .cpp	X		