Programação Orientada por Objetos 2023/2024

Ficha de Laboratório #1 - Parte 2

> ### Objetivos
>
> - Revisões sobre a composição de classes e coleções: revisões.
> - Introdução à utilização do GitHub Classroom.

Programas

Pretende-se desenvolver um programa que permita jogar o tradicional jogo do enforcado.

Regras de implementação

- Criar a aplicação utilizando o IDE BlueJ.
- Implementar o código necessário e testar no fim de cada nível.
- **Atualizar a versão do programa no repositório no mínimo no fim de cada nível** (pode optar por submeter a versão no final de cada alínea).
- Use as convenções de codificação adotadas para a linguagem Java (ver **Notas**).

Implementação

Nível 1:

- Implemente a classe `WordGuessingGame`. Esta classe deverá ter como atributos:
  - `hiddenWord` – representa a palavra que se pretende adivinhar, deve ser inicializada com "abc";
  - `guessedWord` – representa a palavra que se vai adivinhando, deve ser inicializada com `"___"`, três caracteres `"_"`;
  - `numberOfTries` – para contabilizar o número de tentativas.
- Crie o construtor da classe `WordGuessingGame`
- Crie os métodos seletores da classe `WordGuessingGame` para os atributos `hiddenWord`, `guessedWord` e `numberOfTries`
- Crie o método `showGuessedWord` que escreve para o ecrã a palavra que se vai adivinhando.

Nível 2:

- Inclua a classe `InputReader` fornecida com este enunciado no projeto onde está a trabalhar. Esta classe irá permitir ler o texto que o utilizador escrever no teclado. Adicione um atributo `reader`, da classe `InputReader`, à classe `WordGuessingGame`.
- Altere os construtores tendo em conta o novo atributo `reader`
- Crie o método `play` na classe `WordGuessingGame` que lê as letras que o utilizador vai escrevendo até que ele adivinhe a palavra escondida; este método utiliza os seguintes **métodos privados**, que

também deverão ser criados:

  - `showWelcome` - apresenta uma mensagem inicial de boas vindas;
  - `showGuessedWord` – criado no nível anterior;
  - `guess` - analisa se a letra fornecida pelo utilizador está correta (contida em `hiddenWord` e ainda não descoberta) e nesse caso atualiza o atributo `guessedWord`
  - `showResult` - apresenta o número de tentativas, após a palavra ter sido adivinhada.

Nível 3:

- Implemente a classe `WordGenerator`. Esta classe deverá ter como atributo `ArrayList words`
- Crie o método privado `fillArrayList` que adiciona ao `ArrayList` as seguintes palavras (que são keywords de Java): "boolean", "break", "byte", "case", "char", "class", "continue", "do", "double", "else", "enum", "for", "if", "import" e "int".
- Crie o método `generateWord` que gera aleatoriamente um valor de índice do `ArrayList` e devolve a palavra que se encontra nesse índice.
- Crie o método `addWord`, que permite adicionar uma palavra ao `ArrayList`

Nível 4:

- Para completar a classe `WordGuessingGame` comece por acrescentar-lhe um atributo do tipo `WordGenerator`que deverá ser inicializado no construtor. Utilize este atributo para gerar a palavra escondida (`hiddenWord`).
- Para inicializar a palavra que se vai adivinhando (`guessedWord`) crie um método `initializeGuessedWord` que deverá criar uma palavra com a mesma dimensão da palavra escondida (`hiddenWord`) composto apenas por caracteres “_” (sublinhado). Adapte o código para utilizar este método.
- Para finalizar, o método `guess` deverá substituir a letra recebida do utilizador nos locais onde esta apareça na palavra escondida no texto da palavra que se vai adivinhando. Isto apenas no caso de a letra existir na palavra escondida e desde que não tenha sido fornecida antes.
- Teste o jogo para confirmar que está a funcionar.

Nível 5:

- Implemente a classe `FullGame`, que vai permitir que o utilizador adivinhe mais palavras sem terminar o jogo. Esta classe deverá ter como atributos um objeto da classe `WordGuessingGame`, e um objeto da classe `InputReader`.
- Crie o método `reset` da classe `WordGuessingGame` para reinicializar os atributos `hiddenWord`, `guessedWord`, e `numberOfTries`.
- Crie o método `play` da classe `FullGame` que vai utilizar ciclicamente os métodos `reset` e `play` da classe `WordGuessingGame`, enquanto o jogador quiser continuar a jogar (s ou S).

Notas:

Para os identificadores siga as convenções adotadas normalmente, em particular:

1. A notação **camelCase** para o nome das variáveis locais e identificadores de atributos e métodos.
2. A notação **PascalCase** para os nomes das classes.
3. Não utilize o símbolo ‘_’, nem abreviaturas nos identificadores.