**Título:** HW.2- Revisão de código

**Autor:** 65767 João Rafael Duarte de Almeida

**Assunto:** Revisão do código de uma aplicação produzida por outro programador

**Data:** 2 de Março de 2015

Trabalho efetuado no âmbito da disciplina de

Teste e Qualidade do Software

da Licenciatura em Engenharia Informática

Docente(s) responsável(eis)

Prof. Ilídio Oliveira

Após analisar o projeto, foi possível verificar alguns aspetos que não são avaliadas diretamente usando os *Code smells* mas que é importante ter em conta.

Um dos aspetos é que o projeto não realiza nenhuma opção se for executado e as funções que deveria executar no *main* estão vazias.

A análise usando os paramentos do *Code smells* nem sempre pode ser muito conclusiva, pois alguns critérios não são aplicáveis neste projeto.

**Duplicated Code**: Não tem código duplicado, existe código similar na função de “encriptarAES” e “decriptarAES”, mas de modo a facilitar o uso das mesmas, estando separadas poderá ser uma boa política, mas não a mais correta.

**Long Method:** Os métodos implementados são curtos e de fácil compreensão.

**Large Class:** As classes não são exaustivas, e não possuem código redundante nas mesmas. Cada classe tem apenas o que é necessário ter.

**Large Parameter List:** A quantidade de parâmetros de todas as funções é curta.

**Divergent Change:** Na classe *Seguranca*, existem a função de cifra que poderia receber um array de *bytes* em vez de uma *String* para pode ser reutilizada mais facilmente. Assim como a função de decifra poderia retornar um array de *bytes* em vez de uma *String.*

**Feature Envy:** A função *getPass* invoca uma outra função usada noutra classe. Caso a função fizesse o suposto, deveria estar na classe *Utilizador.*

**Data Clumps:** Os dados do utilizador estão juntos numa classe.

**Lazy Class:** A classe *FicheiroDados* apesar de não estar completa, nesta situação poderá não compensar a sua existência.

**Speculative Generality:** Neste caso a classe *FicheiroDados* e *Seguranca* não são usados para nada, mas devido ao facto do projeto estar incompleto, por isso este “code smell” pode não ser aplicável.

**Data Class:** Aplicável no caso da classe *Utilizador*, este apenas tem “gets” e uma função que invoca outra que está noutra classe. Poderia ser organizado de outro modo a que o a classe *Utilizador* tivesse mais funções relacionadas com o objeto em questão.

**Comments:** Todos os comentários usados não estão lá a fazer nada, as funções por si só já são explicativas.

Devido a ser um projeto pequeno, houve “code smells” que não foram mencionados, pois não eram aplicáveis em nenhuma situação.

Com base nas pistas fornecidas no enunciado, e usando o livro “Clean Code” pode-se analisar o código mais pormenorizadamente. Os nomes das entidades, em quase todas as situações, são autoexplicativas, ou seja, pelo seu nome consegue-se perceber qual a sua utilidade, mas existe algumas variáveis que mesmo assim poderiam ser mudadas como a utilização de “f” como nome de uma variável do tipo *File.*

Os métodos até fazem de menos pois estão incompletos, como já foi referido, e o código não é robusto.

Após esta análise ainda foi possível verificar alguns erros no código, que iriam provocar erros de execução. Alguns destes erros são:

* O switch usado no *main* para selecionar a opção a realizar não tem um default, logo se o utilizador usa se um campo não previsto a aplicação daria erro.
* O while usado no *main* com a condição *true*nunca permitiria que a aplicação termina-se, a não ser que fosse usado um “System.exit()” na função sair().
* A função *encriptarAES* contem o argumento frase do tipo String que se fosse alterado para o tipo byte[], tornaria esta função portável para cifrar outros tipos de dados de forma mais fácil.
* O mesmo acontece com a função *desincriptarAES,* se o tipo de *return*fosse byte[], tornaria esta função portável para decifrar outros tipos de dados de forma mais fácil.
* A função que lê o ficheiro não está correta, pois não se pode inicializar o array de bytes com o tamanho 100, sendo que o ficheiro facilmente pode ter um tamanho superior.