PUC-Rio
Data: 04/04/2019
Aluno: Wellington Bezerra Chaves
Matrícula: 1413383

Teste 5

1)Vimos, novamente, os conceitos de acoplamento e coesão. Perguntei como o conceito inventado por David Parnas se chama. Diga qual o nome desse conceito, o que ele significa e como se relaciona com acoplamento e coesão. Justifique sua resposta.

O Desenvolvimento de sistemas sem uma metodologia para a construção de programas, geralmente resulta em um software com vários erros e com alto custo de desenvolvimento que, consequentemente exige um custo elevado para sua correcção e manutenção futuras.

David Parnas criou o conceito de desenvolvimento modular e os fundamentos do que atualmente se conhece por programação orientada a objeto.

Desenvolvimento modular é um conceito onde o sistema ou software é dividido em partes distintas. A modularização de programas juntamente com outras técnicas de programação integram o ferramental para a elaboração de programas visando, principalmente, os aspectos de confiabilidade, legibilidade, manutenção e flexibilidade.

Coesão está, na verdade, ligado ao princípio da responsabilidade única e realizá-la de maneira satisfatória, ou seja, uma classe não deve assumir responsabilidades que não são suas .

O acoplamento significa o quanto uma classe depende da outra para funcionar. E quanto maior for esta dependência entre ambas, dizemos que estas classes elas estão fortemente acopladas.

Um desenvolvimento não modular que ignora o princípio de coesão e com alto acoplamento gera grandes problemas, como dificuldade de manutenção e de reuso.

2) Procure entender o código da página 22 da aula 9, explique seu entendimento ou dificuldade de fazê-lo, usando os conceitos de S,R -| R

Se um pedido de retirada ou deposito for realizado emu ma conta que não existe, então uma nova conta deverá ser criada.

Em um função de retirada, a quantia a ser retirada somente será debitada se for menor igual ao valor do saldo.

Em uma função de depósito, uma quantia será adicionada ao valor do saldo, não havendo restrições.

3) Melhore a organização do código da página 23 da aula 9, reescrevendo-o (tanto a parte referente as anotações (ETVX), como o próprio código, se achar necessário).

```
-- Recebe uma string onde todos os caracteres não-alfanuméricos foram substituídos por
espaços para a função recebida

-- PRE:
str_data é uma string não nula (Verificação: existe uma assertiva garantindo isto)
func é uma função não nula (Verificação: existe uma assertiva garantindo isto)

-- POS: as letras maiúsculas foram substituídas por letras minúsculas - gsub('%W',' '). O
resultado dessa substituição foi passado para a função recebida.

-- em espaços em branco, e lower() transforma os caracteres maiúsculos em minúsculos)


function filter_chars_and_normalize(str_data, func)

     assert(str_data ~= nil, "A string str_data esta nula no comeco da funcao
filter_chars_and_normalize")

     assert(func ~= nil, "A função func esta nula no comeco da funcao
filter_chars_and_normalize")

     func(str_data:gsub('%W',' '):lower(), remove_stop_words)

end
```