**Laboratório 3**



3.8.1 Objetivo:

Além dos objetivos definidos nos laboratórios anteriores temos também o conceito de encapsulamento, criação de array, criação e utilização do método construtor, uso das classes wrappers e a persistência e recuperação das informações em arquivos. No capóitulo 4 iremos abordar com maiores detalhes sobre as classes wrappers.

3.8.2 Definição:

Fazer uma classe Java chamada Lab03Sistema no pacote view e copiar o conteúdo da classe Lab02Sistema. Alterar onde aparece Lab02Sistema para Lab03Sistema. Fazer uma classe Java chamada Lab03ContaCorrente no pacote model e copiar o conteúdo da classe Lab02ContaCorrente. Alterar onde aparece Lab02ContaCorrente para Lab03ContaCorrente. Transformar os atributos da classe Lab03ContaCorrente para privados.

Escrever um construtor na classe Lab03ContaCorrente que receba o número da agência e o número da conta. Este será um construtor com 2 parâmetros do tipo primitivo int (agencia, conta). Em seguida atribuir estes valores recebidos como parâmetros aos atributos da classe Lab03ContaCorrente. Após a atribuição realizar a leitura dos dados de um arquivo. A leitura deverá ser implementada pelo método recuperar () que deverá ser definido como privado, pois não existe a necessidade deste método ser visível a outras classes a não ser para a classe Lab03ContaCorrente. No método recuperar () apresentamos como recuperar um arquivo do disco rígido. No capítulo 8 iremos abordar em detalhes sobreas classes que manipulam streams. Também neste método estamos utilizando o tratamento de exceções. No capítulo 7 iremos abordar sobre o tratamento de exceções.

3.8.2.1 Método recuperar ():

 Listagem de programa 14 – Método recuperar (). Usar para ler dados do disco rígido.

**private** **void** recuperar (){

FileReader tArq1;

BufferedReader tArq2;

**int** tQtde = 4;

**try**

{

// Operação I - Abrir o arquivo

tArq1 = **new** FileReader (getNumAge() + "." + getNumConta() + ".dat");

tArq2 = **new** BufferedReader (tArq1);

// Operação II - Ler atributo/valor e colocar na matriz

String [] tLinha = **new** String [tQtde];

**for** (**int** i = 0; i < tQtde; i++){

tLinha [i] = tArq2.readLine();

}

// Operação III - Fechar o arquivo

tArq2.close();

setNumAge(Integer.*parseInt*(tLinha [0]));

setNumConta(Integer.*parseInt*(tLinha [1]));

setNome(tLinha [2]);

setSaldo(Double.*parseDouble*(tLinha [3]));

}

**catch** (IOException tExcept)

{

tExcept.printStackTrace();

}

}

OBS.: O arquivo contendo os dados da conta corrente deve ser aberto no momento da criação do objeto da classe Lab03ContaCorrente utilizando o construtor com dois parâmetros (agência e conta). Escrever também um construtor que receba todos os atributos como parâmetro e inicialize os atributos privados com estes valores. Este será um construtor com 4 parâmetros, sendo 2 inteiros (agencia, conta), 1 real (saldo) e 1 no formato String (nome).

Para facilitar o trabalho de Entrada/Saída, gravar para cada conta um arquivo, construindo o nome usando o número da agência e da conta no formato "AAA.NNNN", onde "AAA" é o número da agência e "NNNN" é o número da conta. Usar a operação de + permitida pela classe String para montar o nome do arquivo.

Gravar as informações com o método gravar (). Este método deve pertencer a classe Lab03ContaCorrente e deverá ser definido como público. Invocar o método gravar () no método execCadastramento () após a confirmação ser igual a sim. Este método também deverá ser executado após a confirmação de um saque ou depósito. Utilizar o método gravar () apresentado a seguir:

3.8.2.2 Método gravar ():

 Listagem de programa 15 – Método gravar (). Usar para gravar os dados no disco rígido.

public boolean gravar ()

{

FileWriter tArq1;

PrintWriter tArq2;

try

{

// Operação I - Abrir o aquivo

tArq1 = new FileWriter (getNumAge() + "." + getNumConta() + ".dat");

tArq2 = new PrintWriter (tArq1);

tArq2.println (getNumAge());

tArq2.println (getNumConta());

tArq2.println (getNome());

tArq2.println (getSaldo());

// Operação II - Fechar o arquivo

tArq2.close();

return true;

}

catch (IOException tExcept)

{

tExcept.printStackTrace();

return false;

}

}

Alterar o programa Lab03Sistema para considerar os construtores. É importante observar que neste exercício precisamos remover o atributo do tipo da classe Lab03ContaCorrente da classe Lab03Sistema e passar a definir um objeto em cada método da classe Lab03Sistema, todas as vezes que uma operação de cadastramento, saque ou depósito for confirmada. É importante observar que agora teremos um objeto sendo criado após a confirmação, para cada ação executada (cadastramento, saque, deposito ou impressão).

No caso do método execCadastramento (), ao criar o objeto da classe Lab03ContaCorrente executar o construtor com os 4 parâmetros. Para os métodos execSaque (), execDeposito () e execConsulta () usar o construtor com os dois parâmetros (agência e conta), todas as vezes que a operação for confirmada.

OBS.: Não mais setar os atributos do objeto através dos métodos de acesso (setXXX). Esta atribuição agora irá ocorrer através dos construtores com dois ou quatro parâmetros, dependendo da transação a ser realizada.

Nos métodos de acesso (setXXX) dos atributos número da agência e número da conta implementar uma validação para que estes não recebam valores negativos. Para o número da agência não permitir valores maior que 9999 e menor que 1. Para o atributo número da conta não permitir valores maior que 9999999 e menor que 1. Caso os métodos de acesso recebam valores inválidos emitir uma mensagem na tela indicando erro nos parâmetros e voltar a pedir o número novamente. O ideal para esta validação seria a utilização de Exceptions, porém este assunto será abordando a seguir. Em um momento oportuno voltaremos a este exercício e faremos a alteração.

3.8.2.3 Sugestões:

Formatar o saldo da conta que esta sendo impresso no método imprimir () utilizando as classes NumberFormat e DecimalFormat. Vejamos um exemplo:

NumberFormat formatter;

formatter =

DecimalFormat.getCurrencyInstance(**new** Locale("pt","BR"));

formatter.setMinimumFractionDigits(2);

System.out.println ("Saldo : " + formatter.format(saldo));

Não esquecer de gerar os métodos de acesso getXXX () e setXXX (). Os atributos privados deverão ser acessados através dos métodos de acesso.