**Universidade Estadual da Paraíba**

**Centro de Ciência e Tecnologias**

**Departamento de Computação**

**Professora:** Luciana Gomes

**Disciplina:** Linguagem de Programação 2   
**Período:** 2018.1

# Laboratório 03

Neste laboratório iremos experimentar conceitos de orientação a objetos iniciais: classes, objetos e construtores.

## 1a Parte: Entendendo melhor a Linguagem Java

1. Modele uma conta. A ideia aqui é apenas modelar, isto é, só identifique que informações são importantes. Desenhe no papel tudo o que uma Conta tem e tudo o que ela faz. Ela deve ter o nome do titular ( String ), o número ( int ), a agência ( String ), o saldo ( double ) e uma data de abertura ( String ). Além disso, ela deve fazer as seguintes ações: saca, para retirar um valor do saldo; deposita, para adicionar um valor ao saldo; calculaRendimento, para devolver o rendimento mensal dessa conta.
2. Transforme o modelo acima em uma classe Java. Teste-a, usando uma outra classe que tenha o main . Você deve criar a classe da conta com o nome Conta , mas pode nomear como quiser a classe de testes, contudo, ela deve possuir o método main . A classe Conta deve conter pelo menos os seguintes métodos:

* saca que recebe um valor como parâmetro e retira esse valor do saldo da conta
* deposita que recebe um valor como parâmetro e adiciona esse valor ao saldo da conta
* calculaRendimento que não recebe parâmetro algum e devolve o valor do saldo multiplicado por 0.1

Um esboço da classe:

class Conta {

double saldo;

// seus outros atributos e métodos

void saca(double valor) {

// o que fazer aqui dentro?

}

void deposita(double valor) {

// o que fazer aqui dentro?

}

double calculaRendimento() {

// o que fazer aqui dentro?

}

}

Um esboço da classe que possui o main :

class TestaConta {

public static void main(String[] args) {

Conta c1 = new Conta();

c1.titular = "Hugo";

c1.numero = 123;

c1.agencia = "45678-9";

c1.saldo = 50.0;

c1.dataDeAbertura = "04/06/2015";

c1.deposita(100.0);

System.out.println("saldo atual:" + c1.saldo);

System.out.println("rendimento mensal:" + c1.calculaRendimento());

}

}

Faça outros testes, imprima outros atributos e invoque os métodos que você criou a mais. Lembre-se de seguir a convenção java, isso é importantíssimo. Isto é, preste atenção nas maiúsculas e minúsculas, seguindo o seguinte exemplo: nomeDeAtributo , nomeDeMetodo , nomeDeVariavel , NomeDeClasse , etc...

1. Crie um método recuperaDadosParaImpressao() , que não recebe parâmetro mas devolve o texto com todas as informações da nossa conta para efetuarmos a impressão. Dessa maneira, você não precisa ficar copiando e colando um monte de System.out.println() para cada mudança e teste que fizer com cada um de seus funcionários, você simplesmente vai fazer:

Conta c1 = new Conta();

// brincadeiras com c1....

System.out.println(c1.recuperaDadosParaImpressao());

O esqueleto do método ficaria assim:

class Conta {

// seus outros atributos e métodos

String recuperaDadosParaImpressao() {

String dados = "Titular: " + this.titular;

dados += "\nNúmero: " + this.numero;

// imprimir aqui os outros atributos...

// também pode imprimir this.calculaRendimento()

return dados;

}

}

1. Construa duas contas com o new e compare-os com o == . E se eles tiverem os mesmos atributos? Para isso você vai precisar criar outra referência:

Conta c1 = new Conta();

c1.titular = "Danilo";

c1.saldo = 100;

Conta c2 = new Conta();

c2.titular = "Danilo";

c2.saldo = 100;

if (c1 == c2) {

System.out.println("iguais");

} else {

System.out.println("diferentes");

}

1. Crie duas referências para a **mesma** conta, compare-os com o == . Tire suas conclusões. Para criar duas referências para a mesma conta:

Conta c1 = new Conta():

c1.titular = "Hugo";

c1.saldo = 100;

Conta c2 = c1;

O que acontece com o if do exercício anterior?

1. Em vez de utilizar uma String para representar a data, crie uma outra classe, chamada Data. Ela possui 3 campos int , para dia, mês e ano. Faça com que sua conta passe a usá-la. (é parecido com o último exemplo da explicação, em que a Conta passou a ter referência para um Cliente ).

class Conta {

Data dataDeAbertura; // qual é o valor default aqui?

// seus outros atributos e métodos

}

class Data {

int dia;

int mes;

int ano;

}

Modifique sua classe TestaConta para que você crie uma Data e atribua ela a Conta :

Conta c1 = new Conta();

//...

Data data = new Data(); // ligação!

c1.dataDeAbertura = data;

Faça o desenho do estado da memória quando criarmos um Conta .

1. Modifique seu método recuperaDadosParaImpressao para que ele devolva o valor da dataDeAbertura daquela Conta :

class Conta {

// seus outros atributos e métodos

Data dataDeAbertura;

String recuperaDadosParaImpressao() {

String dados = "\nTitular: " + this.titular;

// imprimir aqui os outros atributos...

dados += "\nDia: " + this.dataDeAbertura.dia;

dados += "\nMês: " + this.dataDeAbertura.mes;

dados += "\nAno: " + this.dataDeAbertura.ano;

return dados;

}

}

Teste-o. O que acontece se chamarmos o método recuperaDadosParaImpressao antes de atribuirmos uma data para esta Conta ?

1. O que acontece se você tentar acessar um atributo diretamente na classe? Como, por exemplo:

Conta.saldo = 1234;

Esse código faz sentido? E este:

Conta.calculaRendimento();

Faz sentido perguntar para o esquema da Conta seu valor anual?

1. Crie um método na classe Data que devolva o valor formatado da data, isto é, devolva uma String com "dia/mes/ano". Isso para que o método recuperaDadosParaImpressao da classe Conta possa ficar assim:

class Conta {

// atributos e metodos

String recuperaDadosParaImpressao() {

// imprime outros atributos...

dados += "\nData de abertura: " + this.dataDeAbertura.formatada();

return dados;

}

}

## 2a Parte: Exercício Proposto

Crie uma classe Retangulo. A classe tem os atributos comprimento e largura, cada um dos quais é configurado com o padrão 1. A classe deve ter métodos que calculam o perímetro e a área do retângulo. A classe tem os métodos get e set para o comprimento e a largura. Os métodos set devem verificar se comprimento e largura são, cada um, números de ponto flutuante maiores que 0.0 e menores que 20.0. Escreva uma classe que teste a classe Retangulo.