INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA BAHIA Campus Irecê	**Instituto Federal de Educação, Ciência e Tecnologia da Bahia – *Campus* Irecê** **Disciplina: Linguagem Técnica I** **Prof° Jonatas Bastos**

Nome: _____

LISTA DE EXERCÍCIO 10 – Java.Lang

1 - Crie uma classe TestaInteger e vamos fazer comparações com Integers dentro do main:

```
 Integer x1 = new Integer(10);

Integer x2 = new Integer(10);

if (x1 == x2) {
    System.out.println("igual");
} else {
    System.out.println("diferente");
}
```

E se testarmos com o equals?O que podemos concluir? Como verificar se a classe Integer também reescreve o método toString? Amaioria das classes do Java que são muito utilizadas terão seus métodos equals e toString reescritos convenientemente. Aproveite e faça um teste com o método estático parseInt, recebendo uma String válida e uma inválida (com caracteres alfabéticos), e veja o que acontece!

2 - Como fazer para saber se uma String se encontra dentro de outra? E para tirar os espaços em branco de uma String? E para saber se uma String está vazia? E para saber quantos caracteres tem uma String? Tome como hábito sempre pesquisar o javadoc! Conhecer a API, aos poucos, é fundamental para que você não precise reescrever a roda!

3 Crie uma classe TestaInteger e vamos fazer comparações com Integers dentro do main:

```
Integer x1 = new Integer(10);

Integer x2 = new Integer(10);

if (x1 == x2) {
    System.out.println("igual");
} else {
    System.out.println("diferente");
}
```

E se testarmos com o equals?O que podemos concluir? Como verificar se a classe Integer também reescreve o método toString?

A maioria das classes do Java que são muito utilizadas terão seus métodos equals e toString reescritos convenientemente. Aproveite e faça um teste com o método estático parseInt, recebendo uma String válida e uma inválida (com caracteres alfabéticos), e veja o que acontece!

4 - Utilize-se da documentação do Java e descubra de que classe é o objeto referenciado pelo atributo out da System. Repare que, com o devido import, poderíamos escrever:

```
// falta a declaração da saída

saida = System.out;
saida.println("ola");
```

A variável saida precisa ser declarada de que tipo? É isso que você precisa descobrir.

5 - Crie e imprima uma referência para Conta (ContaCorrente ou ContaPoupanca, no caso de sua Conta ser abstrata):

```
Conta conta = new ContaCorrente();

System.out.println(conta);
```

O que acontece?

6 - Reescreva o método toString da sua classe Conta fazendo com que uma mensagem mais propícia seja devolvida. Lembre-se de aproveitar dos recursos do Netbeans para isto: digitando apenas o começo do nome do método a ser reescrito e pressionando Ctrl+espaço, ele vai sugerir reescrever o método, poupando você do trabalho de escrever a assinatura do mesmo, e de cometer algum engano.

```
public abstract class Conta {


    private double saldo;

    public String toString() {
        return "esse objeto é uma conta com saldo R$" + this.saldo;
    }

    // restante da classe

}
```

Imprima novamente uma referência a `Conta`. O que aconteceu?

7 - Reescreva o método equals da classe Conta para que duas contas com o mesmo **número de conta** sejam consideradas iguais. Para isso, você vai precisar de um atributo numero. Esboço:

```
abstract class Conta {

   private int numero;

   public boolean equals(Object obj) {

        Conta outraConta = (Conta) obj;

        return this.numero == outraConta.numero;
    }

    // restante    , colocar getter e setter para numero, usando Eclipse!
}
```

Crie uma classe TestaComparacaoConta, e dentro do main compare duas instâncias de ContaCorrente com ==, depois com equals, sendo que as instâncias são diferentes mas possuem o mesmo numero de conta (use o setNumero).

8 - Faça com que o equals da sua classe Conta também leve em consideração a String do nome do cliente a qual ela pertence. Se sua Conta não possuir o atributo nome, crie-o. Teste se o método criado está funcionando corretamente.

9 - Crie a classe GuardadorDeObjetos como visto nesse capítulo. Crie uma classe TestaGuardador e dentro do main crie uma ContaCorrente e adicione-a em um GuardadorDeObjetos. Depois teste pegar essa referência como ContaPoupanca, usando casting. Repare na exception que é lançada:

```
 GuardadorDeObjetos guardador = new GuardadorDeObjetos();

ContaCorrente cc = new ContaCorrente();
guardador.adicionaObjeto(cc);

// vai precisar do casting para compilar!
// use Ctrl+1 para o Eclipse gerar para você
ContaPoupanca cp = guardador.pega(0);
```

Teste também o autoboxing do Java 5.0, passando um inteiro para nosso guardador.

10 - Escreva um método que usa os métodos charAt e length de uma String para imprimir a mesma caractere a caractere, com cada caractere em uma linha diferente.

11- Reescreva o método do exercício anterior, mas imprima a String de trás para a frente. Teste-a para *"Socorram-me, subi no ônibus em Marrocos"* e *"Anotaram a data da maratona"*.

12- Dada uma frase, reescreva essa frase com as palavras na ordem invertida. *"Socorram-me, subi no ônibus em Marrocos"* deve retornar *"Marrocos em ônibus no subi Soccoram-me,"*. Utilize o método split da String para te auxiliar.