# Computação Orientada a Objetos 2010
# Exercícios em Laboratório – Eclipse, Pacotes e Tratamento de Exceções

## PARTE I – SUPER-MINI Tutorial ECLIPSE

### 1. Criando projetos

- Menu **File >> New >> Java Project**
- Dê um nome ao projeto

### 2. Criando pacotes

- Menu **File >> New >> Package**
- Dê um nome ao pacote

### 3. Criando Classes

- Menu **File >> New >> Class**
- Dê um nome à Classe
- Se usar pacote diferente de Defaul, dê nome ao pacote no campo Package

### 4. Executando um projeto

- Menu **Run >> Run**
- Escolha Java Application

**PARTE II- Exercício Graphics**

**1)** Crie um novo projeto Java **LabCOO-1**

**2)** Crie um pacote **graphics** e coloque as classes **Shape**, **Circle** e **Rectangle** dentro desse pacote. Para isso, use as definições de classes descritas no final desse texto. Fique a vontade para acrescentar mais linhas de código a essas implementações.

**3)** Crie, em um outro pacote **demo**, uma classe que funcione como um aplicativo simples, que forneça a área de um círculo ou **retângulo**, de acordo com os valores de parâmetros dados como entrada pelo usuário. Para isso, você deve importar as definições de classes **Circle** e **Rectangle** que estão no pacote **graphics.**

**4)** Deve-se tratar as exceções decorrentes da entrada de dados, que deve ser do tipo inteiro.

========== CÓDIGOS ==========

**a) Classe Shape:**

```
/*** Classe Abstrata para computar a area de uma figura geometrica
* */

public abstract class Shape
{
/**
* Para uma subclasse concreta, implemente o codigo para computar a area
* da figura
*/
public abstract double getArea();

}
```

==========

**b) Classe Rectangle**

```
/** Classe concreta para computar a area de uma figura rectangular:
* conhece a sua largura e e altura e a formula para calcular a sua
area.
**/

public class Rectangle extends Shape
{
private double _altura;
private double _largura;

/**
* Inicializa esse objeto Rectangle com os valores dados para a sua
* largura e altura.
*/
public Rectangle(double largura, double altura)
{
```

```
_altura = altura;
_largura = largura;
}


/**
* retorna a area desse objeto Rectangle.
*/
public double getArea()
{
return _altura * _largura;
}
}
```

==========

## c) Classe Circle

```
/**
* Classe concreta para computar a area de uma figura circular: conhece
* o seu raio e a formula para calcular sua area.
**/

public class Circle extends Shape
{

private double _raio;

/**
* Initializa esse objeto Circle com um raio dado.
**/
public Circle(double raio)
{

_raio = raio;
}

/**
* @retorna a area desse objeto Circle.
*/
public double getArea()
{
return Math.PI * _raio * _raio;
}

}
```

**PARTE III- Exercícios diversos**

**III.1 Corrigir e completar o código abaixo. O programa deve sua terminar a execução com *break* somente depois da execução do try.**

```
package finallyworks;
class MyException extends Exception {
   }
public class Main {
       static int count = 0;

       public static void main(String[] args) {
           while (true) {
               try {
                   // Post-increment is zero first time:
                   if (count++ == 0)
                       throw new MyException();
                   if (count == 2)
                       break; // out of "while"
           }
       }
   }
```

**III.2 Utilize herança para criar uma superclasse de exceção chamada ExceptionA e subclasses de exceção ExceptionB e ExceptionC, em que ExceptionB herda de ExceptionA e ExceptionC herda de ExceptionB. Escreva um programa para demonstrar que o bloco catch para o tipo ExceptionA captura exceções de tipos ExceptionB e ExceptionC.**