UNIVERSIDADE FEDERAL RURAL DO RIO DE JANEIRO
DEPARTAMENTO DE COMPUTAÇÃO
IC805 – Linguagem de Programação 3

TR2 - Trabalho 2 de Avaliação **02/09/2022** Prof. Claver Pari Soto

Data de entrega: 08/09/2022 na hora da aula, impresso

Muito importante: Para os problemas, apresente na impressao o **código** solicitado e também o **relatório detalhado** de como conseguiram resolver os problemas, resaltando o que foi aprendido no processo da solução. Esse relatório não pode ser compartilhado entre os grupos. Alem de colocar o código no relatório impresso, esses códigos deverão ser enviados ao e-mail do professor.

Problema 1	Código: 1,0 pontos
	Relatório: 1,0 pontos

Considere que é necessário conhecer as disciplinas que cada aluno de uma turma está fazendo neste período. Então, por exemplo para nossa turma de LP3 vamos ter uma lista de números de matrícula (exemplo 20210027970) seguido pelo código da disciplina (separados por um espaço) em que cada aluno está matriculado. A listagem pode não ter uma ordem específica.

Por exemplo, se o aluno de matricula 20200034989 estiver matriculado em IC805 e IC598 enquanto o aluno de matricula 20190025958 estiver em IC805, IC242 e IC550, então a lista pode ficar assim:

20200034989 IC598

20190025958 IC805

20190025958 IC242

20200034989 IC805

20190025958 IC550

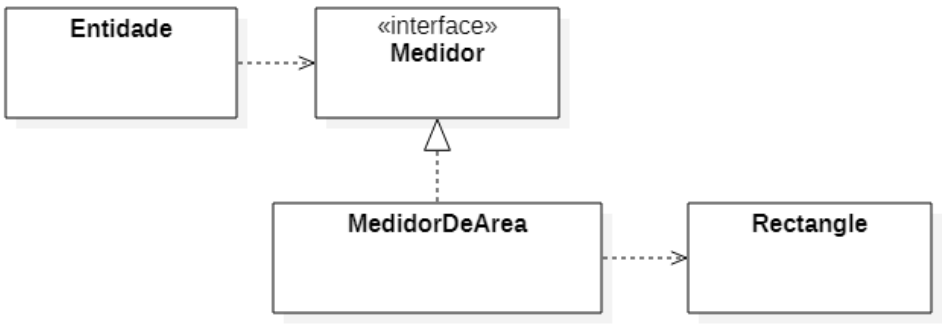
Escreva um programa que permita inserir dados desde o teclado neste formato. Quando o número de matrícula inserida seja -1, então a inserção de dados deve ser detida. Use a classe **`HashMap`** para mapear de um **`Integer`** (o número de matrícula do aluno) para um **`ArrayList`** do tipo **`String`** que contém os códigos das disciplinas que o aluno está matriculado. A declaração deve ficar assim:

```
HashMap<Integer, ArrayList<String>> alunos = new HashMap<Integer, ArrayList<>>();
```

Depois que todos os dados forem inseridos, o programa deve fazer uma iteração no mapa e imprimir no console a matrícula e todos os códigos de disciplinas armazenadas no ArrayList para cada aluno.

Problema 2	Código: 1,0 pontos
	Relatório: 1,0 pontos

Temos aprendido na primeira parte da disciplina, o relacionamento de herança. No diagrama UML abaixo aparecem dois outros tipos de relacionamento. O *relacionamento de dependência* e o *relacionamento de realização (ou implementação)*. A classe **`Entidade`** depende da interface **`Medidor`** e a classe **`MedidorDeArea`** depende da classe **`Retângulo`**. Por outro lado, a classe **`MedidorDeArea`** implementa a interface **`Medidor`**.



Analise e explique detalhadamente cada uma das classes do código da seguinte página. A classe **`Entidade`** (que contém o método **`calcularMedia`**) usa o tipo **`Medidor`** (pois depende dele), mas está desacoplada da

classe **Rectangle**, embora **Entidade** processe objetos tipo **Rectangle**. Também, a classe **Rectangle** não está acoplada a outra classe. Ainda, para processar objetos tipo **Rectangle**, é fornecida uma classe auxiliar **MedidorDeArea**. Esta classe auxiliar tem apenas um propósito: dizer ao método **medir** como medir seus objetos.

```
public class Entidade {
   public static double calcularMedia(Object[] objetos, Medidor medidor) {
      double soma = 0;
      for (Object obj : objetos) {
         soma = soma + medidor.medir(obj);
      }
      if (objetos.length > 0) { return soma / objetos.length; }
      else { return 0; }
   }
}
```

```
public interface Medidor {
   double medir(Object umObjeto);
}
```

```
import java.awt.Rectangle;

public class MedidorDeArea implements Medidor {
   public double medir(Object umObjeto) {
      Rectangle umRetangulo = (Rectangle) umObjeto;
      double area = umRetangulo.getWidth() * umRetangulo.getHeight();
      return area;
   }
}
```

```
import java.awt.Rectangle;
public class MedidorTeste {
   public static void main(String[] args) {
      Medidor medidorArea = new MedidorDeArea();

      Rectangle[] retangulos = new Rectangle[] {
            new Rectangle(5, 10, 20, 30),
            new Rectangle(10, 20, 30, 40),
            new Rectangle(20, 30, 5, 15)
         };

      double areaMedia = Entidade.calcularMedia(retangulos, medidorArea);
      System.out.printf("Média das áreas = %.2f", areaMedia);
   }
}
```

Faça as mudanças necessárias para calcular a média do número de disciplinas matriculadas dos alunos da turma do Problema 1. Reaproveite toda a classe **Entidade** e a interface **Medidor** (não mexa nelas). Crie uma classe **MedidorDeDisciplinas** que implemente **Medidor**, onde o método **medir** retornará o número de disciplinas de um aluno. Você deve criar também uma nova clase cliente (ou teste) muito similar a **MedidorTeste** com a diferença que ela usará uma estrutura contendo as disciplinas dos alunos fornecida pelo programa que você criou para o Problema 1.

0,5 pontos extra

Faça uma pesquisa e explique em alguns parâgrafos e algum exemplo, porque é importante o conhecimento e o uso das interfaces no Java.