Roteiro 4

Parte 1 (roteiro4.parte1) – Classe; Objeto;

Neste roteiro começaremos a trabalhar com os conceitos de Orientação a Objetos.

1 – Crie o pacote **roteiro4.parte1** com as classes **Principal** e **Aluno** conforme o código abaixo.

Atenção :

- A boa prática da programação OO indica que os nomes das classes devem sempre iniciar com letras maiusculas.
- A classe **Aluno** tem apenas os atributos : matrícula, nome, curso, anoIngresso.
- E a classe **Principal** tem o método **main** que utilizamos para **instanciar** o objeto **aluno01,** conforme o destaque em amarelo.

Faça os testes rodando a classe Principal.

```
public class Principal {

    public static void main(String[] args) {

        Aluno aluno01 = new Aluno();
        aluno01.matricula = 111;
        aluno01.nome = "Jose";
        aluno01.curso = "Sistema de Informação";
        aluno01.anoIngresso = 2019;

        System.out.println("Matricula : " + aluno01.matricula);
        System.out.println("Nome : " + aluno01.nome);
        System.out.println("Curso : " + aluno01.curso);
        System.out.println("Ano Ingresso : " + aluno01.anoIngresso);
    }

}
```

```
public class Aluno {

    public int matricula;
    public String nome;
    public String curso;
    public int anoIngresso;
}
```

2 – Faça as devidas implementações na classe Principal para que seja possível instanciar os objetos **aluno01** e **aluno02** . Obs.: Preencha os atributos do aluno02 com dados hipotéticos apenas para teste.

Parte 2 (roteiro4.parte2) – Construtor;

1 – Crie o pacote **roteiro4.parte2** com a cópia das classes **Principal** e **Aluno** implementados na parte1.

Nesta atividade iremos criar o chamado **Método Construtor** da classe Aluno. Ele é o método que permite a criação do objeto. Implemente o método construtor conforme o código abaixo.

Atenção :

- O método construtor deve ter exatamente o mesmo nome da classe.
- Neste método construtor em específico, ele recebe como parâmetro as variáveis : `pMatricula, String pNome, String pCurso, int pAnoIngresso` . E elas são usadas para preencher os respectivos atributos do objeto.

```
public class Aluno {

    public int matricula;
    public String nome;
    public String curso;
    public int anoIngresso;

    Aluno(int pMatricula, String pNome, String pCurso, int pAnoIngresso){
        matricula = pMatricula;
        nome = pNome;
        curso = pCurso;
        anoIngresso = pAnoIngresso;
    }
}
```

2 – Com a implementação do método construtor na classe Aluno podemos instanciar o objeto aluno01 na classe Principal já preenchendo os atributos.

```
public class Principal {

    public static void main(String[] args) {

        Aluno aluno01 = new Aluno(111, "Jose", "SI", 2019);

        System.out.println("Matricula : " + aluno01.matricula);
        System.out.println("Nome : " + aluno01.nome);
        System.out.println("Curso : " + aluno01.curso);
        System.out.println("Ano Ingresso : " + aluno01.anoIngresso);
    }
}
```

Parte 3 (roteiro4.parte3) – Encapsulamento:

1 – Crie o pacote **roteiro4.parte3** com a cópia das classes **Principal** e **Aluno** implementados na parte2.

2 – Na parte2 criamos o objeto aluno01, e no construtor do objeto foi passada todas as informações sobre os atributos deste objeto. Supondo que depois de criado o objeto aluno01, seja necessário modificar o número de matrícula deste aluno. Uma possível solução pode ser implementada conforme o código abaixo.

Faça os devidos testes e verifique se os dados do objeto aluno01 realmente foram modificados.

```
public class Principal {

    public static void main(String[] args) {

        Aluno aluno01 = new Aluno(111, "Jose", "SI", 2019);

        System.out.println("Matricula : " + aluno01.matricula);
        System.out.println("Nome : " + aluno01.nome);
        System.out.println("Curso : " + aluno01.curso);
        System.out.println("Ano Ingresso : " + aluno01.anoIngresso);

        aluno01.matricula = 222;

        System.out.println("Matricula : " + aluno01.matricula);
        System.out.println("Nome : " + aluno01.nome);
        System.out.println("Curso : " + aluno01.curso);
        System.out.println("Ano Ingresso : " + aluno01.anoIngresso);
      }
}
```

3 – A solução indicada no item 2 apesar de funcionar, não é uma boa prática indicada pela POO. O recomendado é que nenhuma classe tenha os seus dados (atributos) acessíveis diretamente. A solução do item 2 só foi possível porque os atributos da classe Aluno são públicos. Para resolver esta questão, modifique o controle de acesso dos atributos da classe Aluno conforme código abaixo. Faça os devidos testes e avalie o impacto desta mudança.

```
public class Aluno {

    private int matricula;
    private String nome;
    private String curso;
    private int anoIngresso;

    Aluno(int pMatricula, String pNome, String pCurso, int pAnoIngresso){
        matricula = pMatricula;
        nome = pNome;
        curso = pCurso;
        anoIngresso = pAnoIngresso;
    }
}
```

4 – A mudança no item 3 provavelmente causou problema na classe Principal, justamente porque os atributos do objeto aluno01 se tornaram inacessíveis. É daí que vem a necessidade do conceito de encapsulamento na POO.

Segue algumas definições para encapsulamento

- O Encapsulamento serve para controlar o acesso aos atributos e métodos de uma classe.
- O Encapsulamento faz com que os atributos/dados de um objeto não fiquem tão facilmente acessíveis. Ou seja, só conseguimos modificar ou acessar os dados de um objeto através de um método.
- Encapsulamento é a técnica que faz com que detalhes internos de uma classe permaneçam ocultos conforme a necessidade e o contexto.

Precisamos criar alguns métodos na classe Aluno que permitam "consultar" (**Get**) um determinado atributo, e outros métodos que permitam "editar" (**Set**) um determinado atributo. Comumente chamamos este conjunto de métodos na POO de **Gets e Sets**.

No código abaixo criamos os Gets e Sets para os atributos matrícula e nome. Crie os métodos correspondentes para os outros atributos.

OBS.: Normalmente as IDEs como Netbeans, VSCode e outras possuem recursos para gerar os Gets e Sets automaticamente. Vale a pena verificar este recurso.

```
public class Aluno {

    private int matricula;
    private String nome;
    private String curso;
    private int anoIngresso;

    Aluno(int pMatricula, String pNome, String pCurso, int pAnoIngresso){
        matricula = pMatricula;
        nome = pNome;
        curso = pCurso;
        anoIngresso = pAnoIngresso;
    }

    public int getMatricula() {
        return matricula;
    }

    public void setMatricula(int matricula) {
        this.matricula = matricula;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }



}
```

5 – Depois de implementar os métodos Gets e Sets na classe Aluno é necessário fazer os devidos ajustes na classe Principal.

```
public class Principal {

    public static void main(String[] args) {

        Aluno aluno01 = new Aluno(111, "Jose", "SI", 2019);

        System.out.println("Matricula : " + aluno01.getMatricula());
        System.out.println("Nome : " + aluno01.getNome());
        System.out.println("Curso : " + aluno01.getCurso());
        System.out.println("Ano Ingresso : " + aluno01.getAnoIngresso());

        aluno01.setMatricula(222);

        System.out.println("Matricula : " + aluno01.getMatricula());
        System.out.println("Nome : " + aluno01.getNome());
        System.out.println("Curso : " + aluno01.getCurso());
        System.out.println("Ano Ingresso : " + aluno01.getAnoIngresso());
      }
}
```

Parte 4 (roteiro4.parte4) – Evoluindo o Projeto:

O nosso pequeno cenário de um projeto de POO evolui como todo projeto de software. Precisamos criar 2 novos atributos na classe Aluno :

- qtdeDisciplinas – um número inteiro para indicar a quantidade de disciplina que o aluno está matriculado.
- situacao – uma string para guarda a informação de quando o aluno está matriculado ou não (Matriculado / Não Matriculado).

Não é desejável que na implementação o aluno tenha a possibilidade de :

- **qtdeDisciplina = 0** e esteja com **situacao = "Matriculado"**
- **qtdeDisciplina = 1** e esteja com **situacao = "Não Matriculado"**

1 – Crie o pacote **roteiro4.parte4** com a cópia das classes **Principal** e **Aluno** implementados na parte3.

2 – Seguindo as boas práticas de OO, crie os 2 atributos indicados acima com o acesso **private.**

3 – Analise e implemente a melhor forma que você acredita que estes 2 novos atributos devam ser contemplados no método construtor da classe Aluno.

4 – Faça as devidas adaptações na classe Principal para que seja possível testar o cenário descrito nesta etapa do projeto.

5 – Faça as adaptações na classe Principal para que os inputs do usuário sejam feitos utilizando o Scanner, e que preencham os dados do objeto aluno01.