MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS APUCARANA
Curso de Bacharelado em Engenharia de Computação

Disciplina: Programação Orientada a Objetos **Turma:** POCO4A – 2021/1

Professor: Lucio Agostinho Rocha

Lista de Exercícios 2 (DUPLA)

---

1. (2,0 pontos) Crie um programa Orientado a Objetos que possua 5 (três) classes: a classe Principal, a classe Computador, a classe Desktop, a classe Celular e a Classe DataCompra. A classe Desktop e a classe Celular são subclasses da classe Computador. A classe Desktop e a classe Celular devem implementar sobrecarga do método toString(). A classe Computador deve conter os atributos nome, modelo e data da compra. A classe Principal deve invocar métodos da seguinte forma: (Dica: veja o exemplo da aula4prog7)

```
Desktop d = new Desktop("comp1");
Celular c = new Celular("cel1");

d.setModelo("Intel").setDataCompra(1,1,2021);
c.setModelo("Samsung").setDataCompra(1,1,2021);
```

2. (2,0 pontos) Crie um programa que utilize Herança para representar a hierarquia de Classes da Figura a seguir:

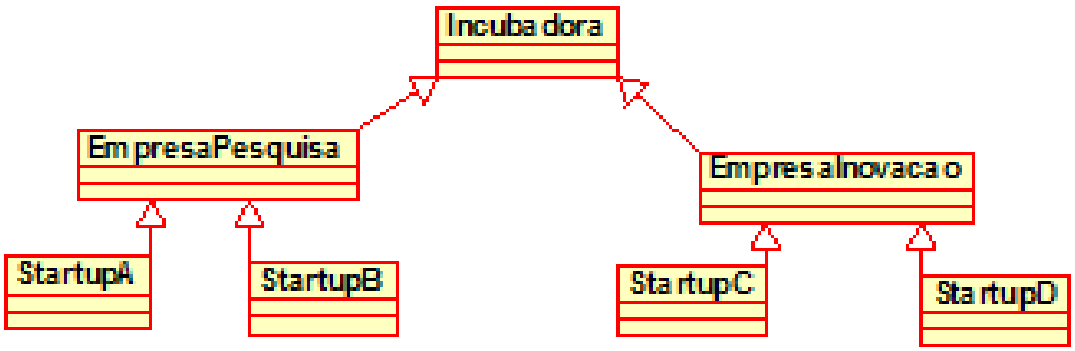

Figura - Diagrama de Classes.

a) (0,4 pontos) Implemente as Classes da hierarquia. Cada Classe possui um método toString que imprime o nome da Classe.

b) (0,4 pontos) Crie um método acessor protected na Classe 'EmpresaPesquisa' que acesse uma variável de instância private da Classe. Invoque esse método na descrição do objeto nas Classes 'StartupA' e 'StartupB'. A descrição do objeto deve

utilizar o método toString para imprimir o nome da Classe e o retorno do método acessor.

c) (0,4 pontos) Crie um método acessor protected na Classe 'EmpresaInovação' que acesse uma variável de instância private da Classe. Esse método deve ser diferente do item a). Invoque esse método na descrição do objeto nas Classes 'StartupC' e 'StartupD'. A descrição do objeto deve utilizar o método toString para imprimir o nome da Classe e o retorno do método acessor.

d) (0,4 pontos) Crie a classe 'Principal' para instanciar e invocar o método toString de objetos das Classes folha da hierarquia, da seguinte forma:

```
StartupA  objA = new StartupA("Minha Nova Empresa");
System.out.println(objA);
```

e) (0,4 pontos) Crie um método acessor e um método mutador nas Classes 'EmpresaPesquisa' e 'EmpresaInovacao'. Utilize chamadas de cada um desses métodos nos objetos criados na Classe 'Principal'.

3. (2,0 pontos) Modifique o exercício 2 para substituir todos os relacionamentos é um (Herança) por relacionamentos tem um (Composição) entre as Classes. A Classe 'Principal' deve instanciar e imprimir objetos da seguinte forma:

```
Incubadora objA = new Incubadora("StartupA");
Incubadora objB = new Incubadora("StartupB", 1);
Incubadora objC = new Incubadora("StartupC", 1, 2);
Incubadora objD = new Incubadora("StartupD", 1, 2, 3);

System.out.println(objA + objB + objC + objD);
```

4. (2,0 pontos) Modifique o exercício 2 da seguinte forma:

a.1) (0,2 pontos) Crie uma nova Classe 'StartupA1' que seja uma subclasse da Classe 'StartupA'.
a.2) (0,2 pontos) A Classe 'EmpresaPesquisa' possui as variáveis de instância privadas 'cnpj' e 'razaoSocial'.
a.3) (0,2 pontos) A Classe 'StartupA' possui as variáveis de instância protected 'tipoAtividade' e 'capitalInicial'.
a.4) (0,2 pontos) A Classe 'StartupA1' possui os métodos acessores privados 'getTipoAtividade' e 'getCapitalInicial'
a.5) (0,2 pontos) Implemente um método toString para exibir com métodos acessores as variáveis de instância herdadas e as declaradas em 'StartupA1'.
Dica: inicialize as variáveis de instância nos métodos Construtores.

b.1) (0,2 pontos) Crie uma nova Classe 'StartupC1' que seja uma subclasse da Classe 'StartupC'.
b.2) (0,2 pontos) A Classe 'EmpresaInovacao' possui as variáveis de instância protected 'nomeInovacao' e 'tipoInovacao'.
b.3) (0,2 pontos) A Classe 'StartupC' acessa as variáveis de instância do item anterior da Classe 'EmpresaInovacao' por meio de herança, sem métodos acessores.

b.4) (0,2 pontos) A Classe 'StartupC1' acessa as variáveis de instância 'nomeInovacao' e 'tipoInovacao' com os métodos acessores privados 'getNomeInovacao' e 'getTipoInovacao'.
a.5) (0,2 pontos) Implemente um método toString para exibir com métodos acessores as variáveis de instância herdadas e as declaradas em 'StartupC1'.
Dica: inicialize as variáveis de instância nos métodos Construtores.

5. (2,0 pontos) Crie um programa Orientada a Objetos com Herança para resolver o seguinte problema:

"Um software de supermercado inicia com uma Tela que recebe como parâmetros um tamanho de tela e uma cor. Uma Tela tem uma Tela de Registro para registrar o nome, o identificador e o preço do alimento. Tela define apenas o tamanho, cor da tela e uma Tela de Registro como atributos. Tela de Registro define apenas o nome do alimento e o identificador do alimento como atributos. Tela de Registro é uma subclasse de Tela de Consulta. Tela de Consulta define apenas o preço do alimento consultado como atributos. "

a) (0,4 pontos) Identifique os nomes que correspondem às Classes. Implemente as Classes.

b) (0,4 pontos) Identifique os verbos que correspondem aos métodos. Implemente os métodos.

c) (0,4 pontos) Identifique os atributos que correspondem às variáveis de instâncias de cada Classe. Declare as variáveis de instância.

d) (0,4 pontos) Implemente os métodos acessores e métodos mutadores para cada variável de instância da Classe.

e) (0,4 pontos) Crie a Classe 'Principal' da seguinte forma:

```
Tela tela = new Tela(100,"azul");
System.out.println(tela.setTelaRegistro("Banana", 1, 3.5f));
System.out.println(tela.getTelaRegistro().getPreco());
```