Centro Federal de Educação Tecnológica –
Disciplina: Programação Orientada a Objetos
Prof. Gustavo Guedes

Aluno(a): _____

Turma:_____ **Data: _____**

Códigos desnecessários e que reduzam o desempenho do sistema serão penalizados.
Utilize as boas práticas de programação, sempre que possível. Vale lembrar que você deve declarar atributos de instância como privados.
LEIA AS QUESTÕES ATÉ O FINAL ANTES DE COMEÇAR.

Questão 1 (6.8) – Desenvolva o código conforme pedido abaixo:

A1 – Um botânico iniciante deseja criar um sistema para buscar as Plantas que estudou e catalogou. Como não sabia Java, catalogou as Plantas em um arquivo txt, uma por linha. Em seguida, pediu para você criar um sistema para buscar as plantas catalogadas. Como você não aprendeu a ler arquivos, encontrou na internet uma classe que lê as linhas de um arquivo e retorna um ArrayList com cada linha do arquivo sendo representada por uma String. Dessa maneira, quando você utiliza a classe LerArquivo e chama o método LerArquivo.retornaStrings("c:/plantas.txt"), o método retorna um ArrayList de Strings no formato id#nome#tamanho#tipo. Note que o retorno do método é List. Nessa questão você precisa criar todas as classes necessárias para o funcionamento, menos a LerArquivo.

Escreva uma classe Planta com 3 atributos: id (String), nome (String) e tamanho (double). Crie os getters e setters apenas se precisar. Crie em Planta APENAS UM construtor, que recebe o id como argumento.

A2 - Implemente um método em uma classe chamada Utils com a seguinte assinatura: public static boolean existe (ArrayList x, Planta y). Escreva esse método de forma que seja verificada a existência do objeto Planta representado por y no ArrayList representado por x, retorne verdadeiro se existir e falso se não existir. Considere que dois objetos Planta são iguais se possuem o **mesmo id**. Não é permitida qualquer iteração para realizar esse item, ou seja, não use *for*, *iterator*, etc.

B – Crie um método em Utils com a seguinte assinatura public static Planta [] ordena (List x). Esse método deve retornar um array com as plantas da Lista x ordenadas pelo tamanho (tanto faz se for em ordem crescente ou decrescete).

C – Ao utilizar o System.out.println em uma referência a um objeto Planta, deve sair no console o id, nome e o tamanho da Planta, ou seja, *a representação desse objeto como String*. Atente para o **idem D.**

D- Crie 2 subclasses da classe Planta: Briofita e Pteridofita. Ao utilizar o System.out.println em uma referência a um objeto Briofita, deve sair no console o texto "[Briofita] " concatenado com id, nome e a tamanho. No caso de Pteridofita, deve sair no console o texto "[Pteridofita] " concatenado com id, nome e a tamanho. Obrigatório **reutilizar** o que foi feito no **item C.**

E - Dada a classe Utils, crie o método public static ArrayList retornaDados(ArrayList stringsPlantas).

Considere que o ArrayList recebido como argumento (stringsPlantas) contém Strings no seguinte formato: id#nome#tamanho#tipo. Por exemplo, considere os elementos desse ArrayList como (D1586#Musgo#15#B, 553-2#Samambaia#4#P, etc.). Esses valores representam id, nome,

tamanho e tipo da Planta (Briofita (B) ou Pteridofita (P)). Em resumo, esse método deve receber um ArrayList contendo Strings e retornar um ArrayList contendo objetos do tipo Briofita ou Pteridofita.

Assim, implemente o método *retornaDados* de forma que seja retornado um novo ArrayList da seguinte forma: os elementos de stringsPlantas devem ser percorridos (lembre-se, são Strings) e os valores do ArrayList que será retornado serão objetos do tipo Briofita ou Pteridofita. Resumindo, você irá criar um objeto Briofita (se o último caractere das Strings em stringsPlantas for B) ou Pteridofita (caso seja P) e adicionar ao ArrayList.

F –Crie uma classe chamada SistemaPrincipal com o método main. Em seguida, receba **do console** o ID de uma planta que deseja verificar se está cadastrada no sistema. Essa classe deve utilizar a classe LerArquivo discutida no início da questão para ler o arquivo de plantas criado pelo botânico. Em seguida, use o método para transformar as Strings lidas pela classe LerArquivo em um ArrayList contendo Briofitas ou Pteridofitas. Desenvolva o resto desse item utilizando os métodos desenvolvidos nos itens anteriores para fazer o seguinte:

(f1) verifique se existe, no ArrayList contendo Briofitas ou Pteridofitas, uma planta com o ID inserido (utilize, obrigatoriamente, o método existente desenvolvido no item A2). Caso exista, exiba, no console, os dados da planta que o botânico cadastrou (id, nome e tamanho), ou seja, imprima *a representação desse objeto como String com base nos itens C e D;* caso tenha dúvidas como proceder, releia o item C e o item Dda questão.

(f1.2) caso não exista uma planta com o ID inserido, exiba no console a seguinte mensagem para o botânico.

A planta com ID XYZ não existe. Considere que XYZ será o id inserido.

Questão 2 (0.4) (FUNRIO - 2014 - IF-BA - Analista de Tecnologia da Informação) Na linguagem Java, um método que é apenas declarado como membro de uma classe, mas não provê uma implementação, deve ser declarado como:

a. abstract.
b. initial.
c. generic.
d. parametrized.
e. void.

Questão 3 (2) – Observe a questão abaixo. Compila? Se sim, O que sai no console?

```
public class Caneca {
	private String cor;
	private int quantidade;
	public static int x;
	public String getCor() {
		return cor;
	}
	public void setCor(String cor) {
		this.cor = cor;
	}
	public int getQuantidade() {
		return quantidade;
	}
	public void setQuantidade(int quantidade) {
		this.quantidade = quantidade;
	}
}
```

```java
public class TesteCaneca {
	public static void main(String[] args) {
		Caneca c = new Caneca();
		c.setCor("verde");
		Caneca c2 = new Caneca();
		c2.setCor("azul");
		c2.setQuantidade(6);
		Caneca c3 = new Caneca();
		c3.setCor("rosa");
		metodoCan1(c2, c3);
		metodoCan2(c3);
		metodoCan3(c);
		c.x = 20;
		c2.x = 15;
		System.out.println(c.x+c2.x+c3.x);
		int i = c3.getQuantidade();
		System.out.println(i);
		System.out.println(c.getQuantidade());
		System.out.println(c2.getQuantidade());
		System.out.println(c3.getQuantidade());
		System.out.println(c.getCor());
		System.out.println(c2.getCor());
		System.out.println(c3.getCor());
		int y = 11;
		c = metodoCan4(c, y);
		System.out.println(c.getCor());
		System.out.println(c.getQuantidade());
		System.out.println(c.x);
		System.out.println(y);
	}
	public static void metodoCan1(Caneca d, Caneca c) {
		d.setCor("bege");
		c.setCor("vermelho");
		c = d;
		c.setQuantidade(9);
		c=null;
	}
	public static void metodoCan2(Caneca c) {
		c.setQuantidade(2);
		c = new Caneca();
		c.setCor("lilás");
	}
	public static void metodoCan3(Caneca c) {
		c = new Caneca();
		c.setCor("prata");
		c.setQuantidade(1);
	}
	public static Caneca metodoCan4(Caneca c, int u) {
		c=new Caneca();
		c.setCor("prata");
		u = u + 15;
		return c;
	}
}
```

Questão 3 (0.4) (IDECAN - 2020 - IF-RR - Informática) A Orientação a Objetos (OO) é um paradigma de programação para o qual "tudo é um objeto", sendo Java uma das principais linguagens que implementam esse paradigma. Em relação à linguagem Java e à OO, analise as seguintes afirmativas: I. Uma classe Java pode implementar mais de uma interface Java. II. Uma classe Java abstrata obrigatoriamente deve possuir um ou mais métodos abstratos. III. Uma classe Java declarada como final não pode ser herdada (não pode ter subclasses Java). Assinale:

a. se todas as alternativas estiverem corretas.
b. se somente as alternativas I e III estiverem corretas.
c. se somente as alternativas I e II estiverem corretas.
d. se somente as alternativas II e III estiverem corretas.
e. se nenhuma das alternativas estiver correta.

Questão 4 (0.4) (CONSULPLAN - 2017 - Câmara de Nova Friburgo - RJ - Oficial Administrativo) Na linguagem de programação Java, o conceito de um objeto ter a capacidade de ser referenciado de diversas formas é conhecido como:

f. Herança.
g. Reescrita.
h. Super Classe.
i. Polimorfismo.