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

Nota

Aluno(a): _____

Turma:_____ **Data:** _____

Esta avaliação deve ser respondida preferencialmente usando caneta esferográfica azul.
Faça distinção clara entre maiúsculas e minúsculas.
Seja claro, formal e sucinto.
Códigos desnecessários e que reduzam o desempenho do sistema serão penalizados.
Utilize as boas práticas de programação.
LEIA AS QUESTÕES ATÉ O FINAL ANTES DE COMEÇAR.

Questão 1 (5,8) – Desenvolva o código conforme pedido abaixo:

A – Escreva uma classe abstrata chamada Pessoa com 3 atributos: cpf, nome e idade (defina os tipos). Crie os getters e setters apenas se precisar. Crie em Pessoa APENAS UM construtor, que recebe o cpf como argumento. Implemente um método em uma classe chamada Utils com a seguinte assinatura: public boolean existe (List<Pessoa> x, Pessoa y). Escreva esse método de forma que seja verificada a existência do objeto Pessoa representado por y na lista representada por x, retorne verdadeiro se existir e falso se não existir. Considere que dois objetos Pessoa são iguais se possuem o **mesmo CPF e a mesma idade**. 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 void ordena (List<Pessoa> x). Esse método deve ordenar os objetos Pessoa em x por ordem alfabética de nome. Prepare a classe Pessoa para que isso ocorra corretamente. Não é permitida qualquer iteração para realizar esse item, ou seja, não use *for*, *iterator*, etc.

C – Ao utilizar o System.out.println em um objeto Pessoa ou filho de Pessoa, deve sair no console o cpf, nome e idade da Pessoa.

D- Crie 2 subclasses da classe Pessoa: PessoaJuridica e PessoaFisica.

E - Dada a classe Utils, crie o método public Map<String, Pessoa> retornaDados(Set<String> conjuntoPessoas) throws FormatoIncorretoException.

Considere que o conjunto recebido como argumento (conjuntoPessoas) contém Strings no seguinte formato: CPF#nome#idade#tipo. Por exemplo, considere os elementos desse conjunto como (080949343-23#Arthur Novaes Moura#34#PF, 310949321-44#Erika da Silva Souza#30#PJ, etc.). Esses valores representam cpf, nome, idade e tipo da Pessoa.

Dessa maneira, implemente o método *retornaDados* de forma que seja retornado um mapa da seguinte forma: os elementos de *conjuntoPessoas* devem ser percorridos e o CPF de cada elemento é a chave do Mapa e os valores do mapa são objetos do tipo PessoaFisica (se o final da String for PF) ou PessoaJuridica (se o final da String for PJ). Resumindo, você irá criar um objeto PessoaFisica ou PessoaJuridica representando cada elemento em *conjuntoPessoas* e adicionar ao mapa. Caso algum elemento em *conjuntoPessoas* possua mais de três caracteres # ou menos de três caracteres #, lance a exceção checked FormatoIncorretoException. O formato de saída da exceção deve ser: FormatoIncorretoException: O formato da String [XXX] esta

incorreto. [XXX] representa a String em conjuntoPessoas que gerou o erro. Crie a classe FormatoIncorretoException como uma exceção checked.

F – Crie um programa principal. Nesse programa, considere que já existe um método denominado BancoDeDados.getPessoas():Set<String> que retorna um conjunto com um milhão de objetos do tipo String apresentado no item E. Podendo utilizar os métodos produzidos nos itens anteriores, utilize o método desenvolvido no item E para retornar um mapa com base no conjunto retornado pelo método BancoDeDados.getPessoas(). Em seguida, imprima no console o cpf, nome e idade de cada pessoa em ordem alfabética.

Dica: Dada uma coleção x (Collection x=...), podemos criar um ArrayList da seguinte maneira: new ArrayList(x). Dessa maneira, todos os elementos da coleção estarão dentro do ArrayList.

Questão 3-a (1,0) – SUPONHA (não a crie AINDA) uma classe Livro com o atributo privado id (String). Crie uma classe denominada CarrinhoDeCompras; crie nela um atributo privado denominado lista que seja uma lista. Crie o get e, no lugar de set, crie um método para inserir um elemento na lista. Crie uma classe principal. Crie um CarrinhoDeCompras. Crie 3 objetos livro recebendo do console o id. Insira no CarrinhoDeCompras os 3 objetos.

Questão 3-b (2,0) Leia a questão até o final. Após ler, você vai criar a classe Livro nesse momento. Crie um método estático em uma classe denominada Utils. Esse método deve receber como argumento um objeto do tipo CarrinhoDeCompras. Utilizando o método frequency de Collections, imprima no console o id e a frequência de cada um dos objetos no CarrinhoDeCompras. Se os objetos Livro tiverem o mesmo id, o id só pode sair uma vez. Ex: vamos considerar que no exemplo da questão 3-a, os ids "123", "123", "333" tenham sido inseridos. A saída no console deverá ser:

"123" - 2

"333" - 1

Questão 4 (1,2) – Observe a questão abaixo. O que sai no console?

```
public class TT2 {
	private static int k;
	private  int j;
	public TT2() {
		j++;
	}
	public static void main(String[] args) {
		TT2 x = new TT2();
		x.k = 55;
		x.j = 33;
		x =  new TT2();
		x.j = 25;
		x.k = 42;
		System.out.println(x.j);
		System.out.println(x.k);
		teste(x.j, x);
		System.out.println(x.k);
		System.out.println(x.j);
		teste2(x.j, x);
		System.out.println(x.j);
		System.out.println(x.k);
		TT2 y = new TT2();
		x =  teste3(x, y);
	}
	public static void teste(int x, TT2 y) {
		y.k = 111;
		x = 91;
	}
	public static void teste2(int x, TT2 y) {
		y = new TT2();
		y.k = 37;
		y.j = 39;
	}
	public static TT2 teste3(TT2 x, TT2 y) {
		x = y;
		TT2 k = x;
		x = new TT2();
		x.j=11;
		return k;
	}
}
```

BOA SORTE!