Quarta lista de exercícios de Programação Orientada a Objetos

Questões teóricas

1. Diferencie shallow copy (cópia rasa) de deep copy (cópia profunda).
2. O que é herança? Qual a vantagem de se usar esse mecanismo?
3. Qual a diferença entre herança simples e herança múltipla? Ambas são suportadas pelo Java?
4. O que é o conceito de polimorfismo? Explique um cenário onde precisamos de polimorfismo.
5. Dado o código abaixo:

```
class Exercicio{
        public String valor;
        public static void main(String[] args){
                Exercicio obj1 = new Exercicio();
                Exercicio obj2 = new Exercicio();
                obj1.valor = “Objeto”;
                ob2.valor = “Objeto”;
                System.out.println(“Os objetos são iguais?” + (obj1 == obj2));
        }
}
```

O que será escrito na tela ao final da execução da função main? Caso imprima false, o que pode ser feito para retornar true?

Questões práticas

Copie os códigos das classes Funcionario, Gerente, Trimestre (enumeração) bem como a Driver Class, todos fornecidos abaixo. Em seguida, responda às perguntas subsequentes.

Classe Funcionario

```
public class Funcionario {
    protected String nome;
    protected int matricula;
    protected double salario;
    protected Funcionario chefe;

    public Funcionario(String n, int m, double s, Funcionario c){
        nome = n;
        matricula = m;
        salario = s;
        chefe = c;
    }

    public double bonus(Trimestre t){
        return salario * 0.1;
    }

    public String getNome(){
        return this.nome;
    }
}
```

Classe Gerente

```
public class Gerente extends Funcionario{
    public Gerente(String nome, int mat, double sal, Funcionario chefe){
        super(nome, mat, sal, chefe);
    }
    @Override
    public double bonus(Trimestre t){
        return (t == Trimestre. 4o) ? salario * 0.15 : bonus(t);
    }
}
```

Classe de Enumeração ***Trimestre:***

```
public enum Trimestre {
    _1o,
    _2o,
    _3o,
     4o
}
```

Driver Class:

```
public class Teste{
    public static void main(String[] args) {

        Gerente jose = new Gerente("José", 2222, 10500.00, null);
        Funcionario eugenio = new Funcionario("Eugênio", 9999, 4500.00,
jose);

        Trimestre t = Trimestre._4o;

        System.out.println("\nBônus de " + jose.getNome() + " no " + t +  "
trimestre:");
        System.out.println(jose.bonus(t));

        System.out.println("\nBônus de " + eugenio.getNome() + " no " + t +
" trimestre:");
        System.out.println(eugenio.bonus(t));
    }
}
```

6. Execute o código. O bônus do Gerente (José) foi de 10 ou de 15%? E o do outro funcionário?
7. Altere, na driver class, o trimestre para que seja o 3º e não mais o 4º e rode novamente. Ocorrerá um erro de execução. Por quê? Corrija o erro **única e exclusivamente pela adição da referência *super* no local apropriado**.
8. Criar o método equals para a classe Funcionario. Dois objetos do tipo Funcionário se tratam do mesmo funcionário se o atributo ***matricula*** é o mesmo em ambos.
9. Modifique a visibilidade de todos os atributos da classe Funcionario para private. O código compila normalmente?
10. Crie os getters e setters públicos necessários para adequar a classe Gerente à nova visibilidade dos atributos de Funcionario.