Exercícios

Observações:

- Os exercícios devem ser feitos individualmente e com o auxílio do material de aula. Evite falar com os colegas nessa lista e usar o dontpad. Para me chamar apenas levante o braço;
- Insira como comentário no código-fonte o nome do desenvolvedor e o enunciado referente a implementação;
- Para cada exercício, crie um pacote com o nome do exercício, por exemplo exercicio3.
- Envie sempre os arquivos de código-fonte (.java);
- Qualquer dúvida, mande e-mail para o professor;
- Todos os métodos e atributos devem ser escritos respeitando o camelCase e o encapsulamento.
- Organize seu tempo e bom trabalho!!!

1. Implemente uma classe **Funcionário** que deve ter o nome do funcionário, o departamento onde trabalha, seu salário (double), a data de entrada no banco (String), seu RG (String) e um valor booleano que indique se o funcionário está na empresa no momento ou se já foi embora.

   Você deve criar alguns métodos de acordo com sua necessidade. Além deles, crie um método bonifica que aumenta o salário do funcionário de acordo com o parâmetro passado como argumento. Crie, também, um método demite, que não recebe parâmetro algum, só modifica o valor booleano indicando que o funcionário não trabalha mais aqui. Identifique que informações são importantes para o funcionário e o que um funcionário faz.

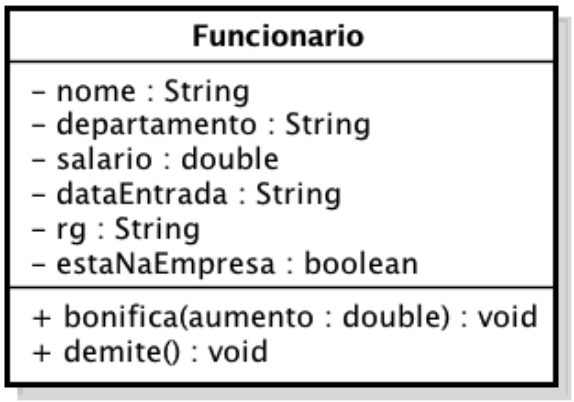
   Um esboço incompleto da classe:

```
class Funcionario {

    double salario;
    // seus outros atributos e métodos

    void bonifica(double aumento) {
          // o que fazer aqui dentro?
       }

    void demite() {
        // o que fazer aqui dentro?
       }
  }
```



2. Crie uma classe para testar a classe **Funcionario** chamada **FuncionarioTeste**. Esta nova classe deve conter o método `main`.

   Um esboço da classe que possui o FuncionarioTeste:

```
public class FuncionarioTeste {

  public static void main(String[] args) {
       testaFuncionario();
  }

  public void testaFuncionario(){
    Funcionario meuFuncionario = new Funcionario();
    //Atribua valores ao funcionário, passando o salario = 1000
    //Execute o método bonifica passando o valor 100
    //Imprima o salario atual
  }

}
```

   Incremente essa classe. Faça outros testes, imprima outros atributos e invoque os métodos que você criou a mais. Teste valores inválidos. Lembre-se de seguir a convenção java, isso é importantíssimo. Isto é, nomeDeAtributo, nomeDeMetodo, nomeDeVariavel, NomeDeClasse, etc...

3. Crie um método mostra() que não recebe nem devolve parâmetro algum, simplesmente imprime a especificação e o valor de todos os atributos do nosso funcionário. Dessa maneira, você não precisa ficar copiando e colando um monte de System.out.println() para verificar o estado do objeto a cada mudança.

```
class Funcionario {
       // seus outros atributos e métodos
```

```
        void mostra() {
            System.out.println("Nome: " + this.nome);
            // imprimir aqui os outros atributos...
            // também pode imprimir this.calculaGanhoAnual()
        }
}
```

Mais tarde veremos o método *toString*, que é uma solução muito mais elegante para mostrar a representação de um objeto como String, além de não jogar tudo pro System.out (só se você desejar).

Na classe de teste, faça o método `testaFuncionario()` executar o recém criado `mostra()`após criar o usuário e bonificá-lo.

```
Funcionario meuFuncionario = new Funcionario();
//código existente
meuFuncionario.mostra();
```

4. Na classe **Funcionario**, construa um método chamado `igual(Funcionario func)` que recebe um funcionários e retorna um valor booleano indicando se o funcionário recebido é igual ao próprio. Na classe de teste, crie um método `testaFuncionariosIguais()` que cria dois novos funcionários (usando *new)* e atribui as variáveis os mesmos valores. Na classe de teste, execute o método `iguais(Funcionario func)` para compará-los.

```
class Funcionario {
  // seus outros atributos e métodos
  boolean iguais(Funcionario func){
    if (this == func) {
        return true;
    } else {
        return false;
    }
}

public class FuncionarioTeste {

  public static void main(String[] args) {
       testaFuncionario();
       testaFuncionariosIguais();
```

```
  }

  public void testaFuncionariosIguais(){
    Funcionario f1 = new Funcionario();
    f1.setNome("Pedro");
    f1.setSalario(100);

    Funcionario f2 = new Funcionario();
    f2.setNome("Pedro");
    f2.setSalario(100);

   if (f1.igual(f2)){
       System.out.println("Os funcionários são iguais");
   } else{
       System.out.println("Os funcionários são diferentes");
   }
  }
}
```

Faça as consistências para executar o método sem erro através da invocação `f1.igual(null)`. Inclua isso no método de teste.

5. Na classe de teste, crie outro método chamado `testaFuncionariosComMesmaReferencia()` que utiliza referências para o mesmo funcionário e compare-os com o método `igual()`. Imprima os resultados de forma semelhante ao método `testaFuncionariosIguais()`. Para criar duas referências para o mesmo funcionário, utilize:

```
Funcionario f1 = new Funcionario():
f1.nome = "Paulo";
f1.salario = 100;
Funcionario f2 = f1;
```

6. Altere a classe **Funcionario** para ao invés de utilizar uma String para representar a data, crie uma outra classe, chamada **Data**, que possui 3 campos int, para dia, mês e ano. Faça com que seu funcionário passe a usá-la.

   Um esboço da classe:

```
class Funcionario {
```

```
    Data dataDeEntrada; // qual é o valor default aqui?
    // seus outros atributos e métodos
}
```

Modifique o método `testaFuncionario()` da classe **FuncionarioTest** para que você crie uma Data e atribua ela ao Funcionario.

```
Funcionario f1 = new Funcionario();
//...
Data data = new Data(); // ligação!
f1.dataDeEntrada = data;
//continua os testes
```

7. Modifique o seu teste para verificar se o nome e o salario são os mesmos: ou seja mesmo com referências diferentes o objeto deve retornar que são iguais:

```
  public void testaFuncionariosIguais(){
     Funcionario f1 = new Funcionario();
     f1.setNome("Pedro");
     f1.setSalario(100);

     Funcionario f2 = new Funcionario();
     f2.setNome("Pedro");
     f2.setSalario(100);

    if (f1.igual(f2)){
        System.out.println("Os funcionários são iguais");//Deve retornar aqui
    } else{
        System.out.println("Os funcionários são diferentes");
    }
   }
 }

  public void testaFuncionariosLevementeDiferentes(){
     Funcionario f1 = new Funcionario();
     f1.setNome("pedro");
     f1.setSalario(100);
```

```
        Funcionario f2 = new Funcionario();
        f2.setNome("Pedro");
        f2.setSalario(100);

       if (f1.igual(f2)){
           System.out.println("Os funcionários são iguais");
       } else{
           System.out.println("Os funcionários são diferentes");
       }
      }

    public void testaFuncionariosLevementeDiferentes2(){
        Funcionario f1 = new Funcionario();
        f1.setNome("Pedro");
        f1.setSalario(100);

        Funcionario f2 = new Funcionario();
        f2.setNome("Pedro");
        f2.setSalario(1000);

       if (f1.igual(f2)){
           System.out.println("Os funcionários são iguais");
       } else{
           System.out.println("Os funcionários são diferentes");
       }
      }
```

8. Modifique seu método `mostra()` para que ele imprima o valor da `dataDeEntrada` daquele funcionário. Para isso, na classe **Data**, crie um método chamado `formatada()` que imprime a data formatada como uma String “dd/mm/aaaa”.

   Crie também um novo método de teste chamado testaDataDeEntrada() que cria um funcionário e executa o método `mostra()` antes e depois de atribuir uma data para este funcionário.

9. Crie uma nova classe chamada **Casa** com a seguinte especificação:
   Classe: Casa

Atributos: cor, porta1, porta2, porta3
Método: void pinta(String s),
int quantasPortasEstaoAbertas()

Crie uma classe chamada **CasaTest** que cria uma casa e pinte-a. Crie três portas e coloque-as na casa; abra e feche as mesmas como desejar. Utilize o método quantasPortasEstaoAbertas para imprimir o número de portas abertas.

10. Crie uma nova classe chamada **Porta** com a seguinte especificação:
Classe: Porta
Atributos: aberta, cor, dimensaoX, dimensaoY, dimensaoZ
Métodos: void abre()
void fecha()
void pinta(String s)
boolean estaAberta()
Em uma classe **PortaTest**, crie uma porta, abra e feche a mesma, pinte-a de diversas cores, altere suas dimensões e use o método estaAberta para verificar se ela está aberta.