

 <p>INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PIAUI</p>	<p>INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ Curso: ADS Disciplina: Programação Orientada a Objetos Professor: Ely</p>
------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Exercício 03

1. Suponha uma classe Hotel que sirva apenas para guardar a quantidade de solicitações de reservas feitas conforme a classe abaixo:

```
public class Hotel {
    int quantReservas;
    void adicionarReserva() {
        quantReservas++;
    }
}
```

Podemos afirmar que haverá um problema de compilação, pois a variável inteira não foi inicializada previamente? Justifique.

2. Ainda sobre a classe do exemplo anterior, considere o método main abaixo:

```
...
public static void main(String args[]) {
    Hotel hotel = new Hotel(2);
    System.out.print(hotel.quantReservas);
}
...
```

Qual o resultado da compilação e/ou execução da classe que tenha o método acima? Justifique.

3. Qual o resultado da execução abaixo. Justifique.

```
public class Teste {
    int b = 1;
    int a;

    Teste(int a) {
        a = b + a;
        System.out.print(this.a);
    }

    public static void main(String args[]) {
        Teste t = new Teste(2);
    }
}
```

4. Dado o seguinte trecho de código de acumulação de um atributo valor, explique a problemática envolvida e reescreva o método com uma solução:

```
void x(double valor) {
    valor = valor + valor;
}
```

5. Considere as classes Radio e TestaRadio abaixo:

<pre>public class Radio { int volume; Radio(int volume) { this.volume = volume; } }</pre>	<pre>public class TestaRadio { public static void main(String[] args) { Radio r = new Radio(); r.volume = 10; } }</pre>
---------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------

Justifique o erro de compilação na classe TestaRadio e proponha uma solução.

6. Considerando o uso da classe Conta abaixo em uma classe com o método main():

```
public static void main(String[] args) {
    Conta c1 = new Conta("1", 100);
    Conta c2 = new Conta("2", 100);
    c1 = c2;
    c1.sacar(10);
    c1.transferir(c2, 50);

    System.out.println(c1.saldo);
    System.out.println(c2.saldo);
}
```

- Qual o resultado dos dois "prints"? Justifique sua resposta.
- O que acontece com o objeto para o qual a referência c1 aponta?

Para as próximas questões: a cada novo método criado nas questões abaixo, crie/altere primeiro teste na classe de testes imprimindo ou comentando um resultado esperado.

7. Crie uma classe chamada Jogador e nela:

- Crie 3 atributos inteiros representando força, nível e pontos atuais;
- Crie um construtor no qual os 3 parâmetros são passados e inicialize os respectivos atributos;
- Crie um método que calcule os pontos relativos a um ataque que são calculados pela multiplicação de força pelo nível;
- Crie um método chamado atacar em que é passado um outro jogador como parâmetro e é feita a subtração de pontos de tal jogador baseado na quantidade de pontos do jogador atual ("this").
- Avalie em uma classe de testes dois jogadores instanciados e inicializados através do construtor. Nessa classe, utilize o método de ataque de cada jogador e ao final, verifique qual jogador tem mais pontos.

8. Altere a classe conta dos slides conforme as instruções abaixo:

- Altere o método sacar de forma que ele retorne verdadeiro ou falso. Caso o saque deixe saldo negativo, o mesmo não será realizado, retornando falso;
- Altere o método transferir() para que o mesmo use os métodos sacar() e depositar(). Visto pelo prisma da "proteção do saldo", chamar outros métodos em vez de acessar o saldo diretamente é mais seguro?

- c. Altere o método `transferir()` para que ele retorne também um valor lógico e que não seja feita a transferência caso o `sacar()` na conta origem não seja satisfeito;
- d. Verifique as diferentes operações implementadas na classe de testes;

9. Crie uma classe chamada `Produto` e nela:

- a. Crie os atributos `codigo`, `descricao`, `valor` e `quantidade`.
- b. Crie os métodos `baixar(int quantidade)` e `repor(int quantidade)` que reduzem e incrementam a quantidade disponível do produto;
- c. Crie um construtor que inicializa todos os atributos;
- d. Crie um atributo `quantidadeMinima` e reescreva o método `baixar` para que não seja possível realizar a baixa caso a operação deixe a quantidade abaixo da quantidade mínima;
- e. Crie um método da classe `Produto` chamado `reajusta(double taxa)` que reajusta em x% o valor do produto.
- f. Crie um método chamado `toString()` que retorna a representação textual do produto concatenando todos os atributos e
- g. Crie um método `equals(Produto produto)` que retorna `true` ou `false` se o produto passado como parâmetro possui o mesmo código;
- h. Verifique as diferentes operações implementadas na classe de testes;

Para pesquisar:

- Pesquise os métodos `toString()` e `equals()` na Internet e veja que eles já existem nos objetos em Java;
- O ideal é que os atributos importantes como `saldo`, `valor`, `pontos` não possam ser alterados com o acesso direto e que eles fossem ocultos do programador e apenas acessados por certos métodos. Com base nisso, pesquise sobre **encapsulamento**.