

Questões – static e final em Java

Parte 1 – Questões objetivas

Questão 1

Em Java, qual é a principal característica de um atributo declarado com a palavra-chave `static`?

- A) Cada objeto da classe possui uma cópia independente do atributo.
 - B) O atributo só pode ser acessado dentro de métodos `static`.
 - C) O atributo pertence à classe e é compartilhado entre os objetos.
 - D) O atributo não pode ter seu valor alterado.
 - E) O atributo só pode armazenar valores constantes.
-

Questão 2

Considere o código:

```
public class Pessoa {  
  
    String nome;  
    static int quantidade = 0;  
  
    public Pessoa(String nome) {  
        this.nome = nome;  
        quantidade++;  
    }  
}
```

Após executar:

```
Pessoa p1 = new Pessoa("João");  
Pessoa p2 = new Pessoa("Maria");  
Pessoa p3 = new Pessoa("Carlos");
```

```
System.out.println(Pessoa.quantidade);
```

Qual será o resultado?

- A) 0
 - B) 1
 - C) 2
 - D) 3
 - E) Ocorre um erro de compilação.
-

Questão 3

Sobre métodos `static`, assinale a alternativa correta.

- A) Um método `static` precisa obrigatoriamente ser chamado através de um objeto.
 - B) Um método `static` pertence à classe e pode ser chamado sem criar um objeto.
 - C) Um método `static` pode acessar diretamente qualquer atributo de instância.
 - D) Um método `static` não pode receber parâmetros.
 - E) Todo método de uma classe deve obrigatoriamente ser `static`.
-

Questão 4

Considere:

```
public class Calculadora {  
    public static int somar(int a, int b) {  
        return a + b;  
    }  
}
```

Qual é a forma mais adequada de chamar o método `somar()`?

A)

```
Calculadora c = new Calculadora();  
c.somar(10, 20);
```

B)

```
Calculadora.somar(10, 20);
```

C)

```
static Calculadora.somar(10, 20);
```

D)

```
new static Calculadora().somar(10, 20);
```

E)

```
Calculadora -> somar(10, 20);
```

Questão 5

Observe:

```
public class Produto {  
    private String nome;  
    private static int totalProdutos;  
  
    public Produto(String nome) {  
        this.nome = nome;  
        totalProdutos++;  
    }  
}
```

Qual é a finalidade mais adequada do atributo `totalProdutos`?

- A) Armazenar um contador específico para cada objeto.
 - B) Armazenar uma informação compartilhada pela classe.
 - C) Impedir a criação de objetos.
 - D) Tornar `nome` uma constante.
 - E) Permitir herança múltipla.
-

Questão 6

Sobre a palavra-chave `final`, assinale a alternativa correta.

- A) Uma variável `final` pode receber novos valores livremente.
 - B) Uma variável `final` deve obrigatoriamente ser `static`.
 - C) Uma variável `final` não pode ter seu valor alterado depois de inicializada.
 - D) Uma variável `final` só pode armazenar números.
 - E) `final` só pode ser utilizada em métodos.
-

Questão 7

Considere:

```
final double PI = 3.14159;
```

```
PI = 3.14;
```

O que acontecerá?

- A) O valor de `PI` será alterado normalmente.
 - B) O programa será executado, mas emitirá um aviso.
 - C) Ocorrerá erro de compilação.
 - D) O valor será alterado automaticamente para `3.14`.
 - E) O Java criará uma nova variável `PI`.
-

Questão 8

Qual declaração representa uma constante de classe seguindo a convenção normalmente utilizada em Java?

A)

```
public int idade = 18;
```

B)

```
public static int idade = 18;
```

C)

```
public final int idade = 18;
```

D)

```
public static final int IDADE_MINIMA = 18;
```

E)

```
constant int IDADE_MINIMA = 18;
```

Questão 9

Considere:

```
public class Configuracao {  
    public static final String VERSAO = "1.0";  
}
```

Qual afirmação está correta?

- A) `VERSAO` pertence a cada objeto individualmente.
 - B) `VERSAO` pode ser alterada depois de inicializada.
 - C) `VERSAO` pertence à classe e não pode ter seu valor alterado.
 - D) É obrigatório criar um objeto de `Configuracao` para acessar `VERSAO`.
 - E) `static final` permite que o valor seja alterado apenas uma vez por objeto.
-

Questão 10

Analise:

```
public class Conta {  
    private double saldo;  
  
    public static void imprimirSaldo() {  
        System.out.println(saldo);  
    }  
}
```

Por que esse código apresenta um problema de compilação?

- A) Porque métodos `static` não podem utilizar `System.out`.
 - B) Porque `saldo` é `private`.
 - C) Porque `saldo` é um atributo de instância e o método `static` não possui uma referência a um objeto específico.
 - D) Porque métodos `static` não podem imprimir valores.
 - E) Porque todo atributo utilizado por um método `static` precisa ser `final`.
-

Parte 2 – final em classes e métodos

Questão 11

Considere:

```
public final class Veiculo {  
}
```

O que significa declarar a classe como `final`?

- A) A classe não pode possuir atributos.
 - B) A classe não pode possuir métodos.
 - C) A classe não pode ser instanciada.
 - D) A classe não pode ser herdada.
 - E) A classe só pode possuir métodos `static`.
-

Questão 12

Considere:

```
public class Animal {  
  
    public final void emitirSom() {  
        System.out.println("Som");  
    }  
}
```

Uma classe filha tenta fazer:

```
@Override  
public void emitirSom() {  
    System.out.println("Outro som");  
}
```

O que acontecerá?

- A) O método será sobrescrito normalmente.
 - B) O método será sobrecarregado.
 - C) Ocorrerá erro de compilação porque métodos `final` não podem ser sobrescritos.
 - D) O Java escolherá automaticamente qual método executar.
 - E) O método da classe pai será transformado em `static`.
-

Questão 13

Qual alternativa apresenta corretamente os três usos principais de `final`?

- A) Variável: impede alteração; método: impede sobrescrita; classe: impede herança.
- B) Variável: impede herança; método: impede instanciação; classe: impede alteração.
- C) Variável: cria objeto; método: cria constante; classe: permite sobrescrita.

- D) Variável: torna o método `static`; método: torna a classe abstrata; classe: impede objetos.
E) Nenhuma das alternativas.
-

Parte 3 – Questões de análise de código

Questão 14

Observe o código:

```
public class Funcionario {  
    private String nome;  
    private static double salarioMinimo = 1500.00;  
  
    public Funcionario(String nome) {  
        this.nome = nome;  
    }  
  
    public static double getSalarioMinimo() {  
        return salarioMinimo;  
    }  
}
```

Responda:

- a) O atributo `salarioMinimo` pertence aos objetos ou à classe?
 - b) É necessário criar um objeto `Funcionario` para chamar `getSalarioMinimo()`?
 - c) Como o método poderia ser chamado?
-

Questão 15

Analise:

```
public class Matematica {  
    public static final double PI = 3.141592;  
  
    public static double calcularArea(double raio) {  
        return PI * raio * raio;  
    }  
}
```

Responda:

- a) Por que `PI` foi declarado como `static`?
 - b) Por que `PI` foi declarado como `final`?
 - c) Qual é a vantagem de utilizar `static final` nesse caso?
-

Questão 16

Analise:

```
public class Pessoa {  
    private String nome;  
  
    public static void mostrarNome() {  
        System.out.println(nome);  
    }  
}
```

O código apresenta um erro.

- a) Explique o motivo do erro.
 - b) Apresente uma solução utilizando um método de instância.
 - c) Apresente uma solução utilizando um parâmetro no método `static`.
-

Gabarito

Questão	Resposta
---------	----------

1	C
2	D
3	B
4	B
5	B
6	C
7	C
8	D
9	C
10	C
11	D
12	C
13	A

Respostas das questões discursivas

Questão 14

- a) `salarioMinimo` pertence à **classe**, pois foi declarado como `static`.
- b) Não. Como o método também é `static`, ele pode ser chamado diretamente pela classe.
- c)

```
double salario = Funcionario.getSalarioMinimo();
```

Questão 15

- a) `static` indica que existe uma única cópia de `PI` associada à classe `Matematica`, e não uma cópia para cada objeto.
- b) `final` impede que o valor de `PI` seja alterado depois de inicializado.
- c) `static final` é adequado para representar uma **constante de classe**. Nesse caso, não é necessário criar um objeto de `Matematica` para utilizar o valor.

Exemplo:

```
System.out.println(Matematica.PI);
```

Questão 16

- a) O atributo `nome` é um atributo de **instância**. Já o método `mostrarNome()` é `static`, portanto pertence à classe e não possui um objeto específico para acessar `nome`.

- b) Uma solução é transformar o método em método de instância:

```
public void mostrarNome() {  
    System.out.println(nome);  
}
```

Uso:

```
Pessoa pessoa = new Pessoa();  
pessoa.mostrarNome();
```

- c) Outra possibilidade é passar o `nome` como parâmetro:

```
public static void mostrarNome(String nome) {  
    System.out.println(nome);  
}
```

Uso:

```
Pessoa.mostrarNome("Fabrício");
```