

Questões — Modificadores de Acesso em Java

Parte 1 — Questões objetivas

Questão 1

Em Java, qual modificador de acesso permite que um atributo ou método seja acessado de **qualquer classe**, independentemente do pacote?

- A) `private`
 - B) `protected`
 - C) `public`
 - D) `default`
-

Questão 2

Considere a classe:

```
public class Pessoa {  
    private String nome;  
}
```

O que significa declarar o atributo `nome` como `private`?

- A) Pode ser acessado diretamente por qualquer classe.
 - B) Pode ser acessado apenas por classes do mesmo pacote.
 - C) Pode ser acessado diretamente apenas dentro da própria classe `Pessoa`.
 - D) Pode ser acessado diretamente somente por subclasses.
-

Questão 3

Quando nenhum modificador de acesso é declarado em um atributo, método ou classe, qual é o nível de acesso utilizado?

- A) `public`
 - B) `private`
 - C) `protected`
 - D) `default` (package-private)
-

Questão 4

Considere:

```
package modelo;
```

```
public class Pessoa {  
    protected String nome;  
}
```

Uma classe pertencente ao **mesmo pacote** de **Pessoa** pode acessar diretamente o atributo **nome**?

- A) Sim.
 - B) Não, pois **protected** permite acesso apenas por subclasses.
 - C) Não, pois somente **public** permite acesso dentro do pacote.
 - D) Sim, mas somente por meio de um método **getNome()**.
-

Questão 5

Qual das alternativas apresenta corretamente a característica do modificador **protected**?

- A) Permite acesso somente dentro da própria classe.
 - B) Permite acesso dentro do mesmo pacote e também por subclasses, respeitando as regras de herança.
 - C) Permite acesso somente dentro do mesmo pacote.
 - D) Permite acesso de qualquer classe.
-

Questão 6

Observe:

```
class Produto {  
    String descricao;  
}
```

Sobre o atributo **descricao**, é correto afirmar que:

- A) É automaticamente **private**.
 - B) É automaticamente **public**.
 - C) Possui acesso **default**, permitindo acesso dentro do mesmo pacote.
 - D) Possui acesso **protected**.
-

Questão 7

Considere as classes:

```
package pacote1;  
  
public class A {  
    private int x;  
    protected int y;  
    public int z;  
    int w;  
}
```

Uma classe B, pertencente ao mesmo pacote `pacote1`, poderá acessar diretamente:

- A) Apenas z.
 - B) y, z e w.
 - C) x, y, z e w.
 - D) Apenas x e z.
-

Questão 8

Considere agora que a classe B esteja em outro pacote:

```
package pacote2;  
  
import pacote1.A;  
  
public class B extends A {  
  
}
```

Dentro de B, qual atributo de A poderá ser acessado diretamente, considerando os modificadores da questão anterior?

- A) x
 - B) y
 - C) w
 - D) Nenhum deles
-

Questão 9

Qual modificador é normalmente utilizado para implementar o princípio de **encapsulamento**, mantendo atributos protegidos contra acesso direto externo?

- A) `public`
 - B) `private`
 - C) `protected`
 - D) `default`
-

Questão 10

Considere:

```
public class Conta {  
  
    private double saldo;  
  
    public double getSaldo() {  
        return saldo;  
    }  
  
    public void depositar(double valor) {
```

```
        saldo += valor;
    }
}
```

Por que o atributo `saldo` foi declarado como `private`?

- A) Para impedir que a própria classe `Conta` acesse o atributo.
 - B) Para permitir acesso direto de qualquer classe.
 - C) Para controlar o acesso ao atributo e promover encapsulamento.
 - D) Para impedir que métodos `public` sejam utilizados.
-

Parte 2 — Questões de análise de código

Questão 11

Analise o código:

```
class Pessoa {
    private String nome;

    public void setNome(String nome) {
        this.nome = nome;
    }

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

Explique por que os métodos `getNome()` e `setNome()` são `public`, enquanto o atributo `nome` é `private`.

Questão 12

Considere:

```
package escola;

public class Aluno {
    String nome;
}
```

E:

```
package escola;

public class Teste {
    public static void main(String[] args) {
        Aluno aluno = new Aluno();
        aluno.nome = "João";
    }
}
```

```
}  
}
```

O código compila? Justifique.

Questão 13

Considere:

```
package modelo;  
  
public class Pessoa {  
    protected String nome;  
}
```

E outra classe:

```
package teste;  
  
import modelo.Pessoa;  
  
public class Teste {  
    public static void main(String[] args) {  
        Pessoa pessoa = new Pessoa();  
        pessoa.nome = "João";  
    }  
}
```

O código compila? Explique considerando que Teste **não é uma subclasse** de Pessoa.

Questão 14

Considere:

```
public class Funcionario {  
    private double salario;  
  
    protected void aumentarSalario(double percentual) {  
        salario += salario * percentual / 100;  
    }  
}
```

Qual é a diferença de acesso entre `salario` e `aumentarSalario()`?

Parte 3 — Questões discursivas

Questão 15

Explique a diferença entre os modificadores:

- `public`
 - `private`
 - `protected`
 - `default`
-

Questão 16

Explique o que significa dizer que um atributo `private` promove **encapsulamento** em programação orientada a objetos.

Questão 17

Explique a diferença entre `protected` e `default`.

Questão 18

Uma classe possui um atributo:

```
private double valor;
```

Um aluno afirma:

"Como o atributo é `private`, nenhuma outra classe poderá utilizar o seu valor."

Essa afirmação está completamente correta? Explique.

Gabarito

Questões objetivas

Questão	Resposta	Justificativa
1	C	<code>public</code> permite acesso a partir de qualquer classe, respeitando a visibilidade da classe e do membro.
2	C	Um membro <code>private</code> pode ser acessado diretamente somente dentro da própria classe.
3	D	A ausência de modificador resulta em acesso default , também chamado de package-private .
4	A	<code>protected</code> permite acesso às classes do mesmo pacote.
5	B	<code>protected</code> permite acesso no mesmo pacote e também por subclasses, inclusive em determinadas situações entre pacotes.
6	C	Sem modificador, o membro possui acesso default , ficando acessível dentro do mesmo pacote.
7	B	No mesmo pacote, <code>protected</code> , <code>public</code> e <code>default</code> podem ser acessados.

Questão	Resposta	Justificativa
		<code>private</code> não.
8	B	Uma subclasse pode acessar um membro <code>protected</code> herdado, mesmo estando em outro pacote, observadas as regras de acesso de <code>protected</code> .
9	B	<code>private</code> é frequentemente utilizado para proteger o estado interno da classe e implementar encapsulamento.
10	C	O <code>private</code> impede o acesso direto externo, permitindo que a classe controle como o atributo é manipulado.

Respostas das questões discursivas

Questão 11 — Resposta esperada

O atributo `nome` é `private` para impedir o acesso direto por outras classes. Os métodos `getNome()` e `setNome()` são `public` para permitir que outras classes consultem ou alterem o valor de forma controlada.

Isso é um exemplo de **encapsulamento**.

Questão 12 — Resposta esperada

Sim, o código compila.

O atributo `nome` não possui um modificador explícito, portanto possui acesso **default (package-private)**.

Como `Aluno` e `Teste` pertencem ao mesmo pacote `escola`, `Teste` pode acessar diretamente `nome`.

Questão 13 — Resposta esperada

Não, o código não compila.

O atributo `nome` é `protected`. Como `Teste` está em outro pacote e não é uma subclasse de `Pessoa`, ela não pode acessar diretamente esse atributo.

Questão 14 — Resposta esperada

O atributo `salario` é `private`, portanto pode ser acessado diretamente somente dentro da própria classe `Funcionario`.

O método `aumentarSalario()` é `protected`, portanto pode ser acessado:

- pela própria classe;
- por classes do mesmo pacote;
- por subclasses, respeitando as regras de acesso de `protected`.

Questão 15 — Resposta esperada

Modificador	Acesso
<code>public</code>	Qualquer classe que tenha acesso à classe/membro
<code>private</code>	Somente dentro da própria classe
<code>protected</code>	Dentro do mesmo pacote e por subclasses, conforme as regras de <code>protected</code>
<code>default</code>	Dentro do mesmo pacote

Uma forma simples de memorizar:

`private` → classe
`default` → pacote
`protected` → pacote + herança
`public` → geral

Questão 16 — Resposta esperada

Encapsulamento significa proteger o estado interno de um objeto e controlar como ele pode ser acessado ou modificado.

Por exemplo:

```
public class Conta {  
    private double saldo;  
  
    public void depositar(double valor) {  
        if (valor > 0) {  
            saldo += valor;  
        }  
    }  
}
```

Nesse caso, uma classe externa não pode fazer:

```
conta.saldo = -1000;
```

O acesso ao saldo é controlado pelo método `depositar()`.

Questão 17 — Resposta esperada

A principal diferença é que:

- **`default`** permite acesso às classes do **mesmo pacote**.
- **`protected`** permite acesso às classes do **mesmo pacote** e também às **subclasses**, respeitando as regras específicas de acesso.

Exemplo:

```
class Pessoa {
```

```
String nome;           // default
protected int idade;   // protected
}
```

Questão 18 — Resposta esperada

Não.

O fato de um atributo ser `private` significa que ele **não pode ser acessado diretamente por outras classes**. Entretanto, a própria classe pode fornecer métodos `public` ou `protected` para permitir acesso controlado.

Por exemplo:

```
public class Produto {
    private double valor;

    public double getValor() {
        return valor;
    }
}
```

Outra classe não pode fazer:

```
produto.valor = 100;
```

mas pode fazer:

```
double valor = produto.getValor();
```

Isso demonstra a aplicação do **encapsulamento**.