## Teoria:

1. Defina:
   a. Sobrecarga de métodos (OVERLOADING)
   b. Sobrescrita de métodos (OVERRIDE)
   c. Métodos abstratos

2. Explique a função de cada palavra reservada em Java:
   a. `interface`
   b. `static`
   c. `implements`
   d. `extends`
   e. `throw`
   f. `throws`

3. Analise o código abaixo, marque a **alternativa correta** e **justifique sua resposta**:

```
class A{
	public final void print() {
		System.out.println("Hello, A!");
	}
}

class B extends A{
	@Override
	public void print() {
		System.out.println("Hello, B!");
	}
}

class Test{
	public static void main(String[] args) {
		B b = new B();
		b.print();
	}
}
```

- O código executa com sucesso e imprime "Hello, A!"
- O código executa com sucesso e imprime "Hello, B!"
- O código não compila
- O código compila, mas ao executar lança exceção

| **Resposta:** | |
|---|---|
| **Justificativa:** | |

4. Analise o código abaixo, marque a **alternativa correta** e **justifique sua resposta**:

```
final class A{
	public final void print() {
		System.out.println("Hello, A!");
	}
}

class B extends A{

}

class Test{
	public static void main(String[] args) {
		B b = new B();
		b.print();
	}
}
```

- O código executa com sucesso e imprime "Hello, A!"
- O código não compila
- O código compila, mas ao executar lança exceção

| **Resposta:** | |
|---|---|
| **Justificativa:** | |

5. Analise o código abaixo, marque a **alternativa correta** e **justifique sua resposta**:

```
final class A{
	public void print() throws Exception {
		System.out.println("Hello, A!");
	}
}

class Test{
	public static void main(String[] args) {
		A a = new A();
		a.print();
	}
}
```

- O código executa com sucesso e imprime "Hello, A!"
- O código não compila
- O código compila, mas ao executar lança exceção

| **Resposta:** | |
|---|---|
| **Justificativa:** | |

6. Analise o código abaixo, marque a **alternativa correta** e **justifique sua resposta**:

```
class A{
	public static void print() {
		System.out.println("Hello, A!");
	}
}

class Test{
	public static void main(String[] args) {
		A a = new A();
		a.print();
	}
}
```

- O código executa com sucesso e imprime "Hello, A!"
- O código não compila
- O código compila, mas ao executar lança exceção

| **Resposta:** | |
|---|---|
| **Justificativa:** | |

7. Analise o código abaixo, marque a **alternativa correta** e **justifique sua resposta**:

```
interface IA{
	void print();
}

class A implements IA{
	@Override
	public void print() {
		System.out.println("Hello, A!");
	}
}

class Test{
	public static void main(String[] args) {
		A a = new A();
		a.print();
	}
}
```

- O código executa com sucesso e imprime "Hello, A!"
- O código não compila
- O código compila, mas ao executar lança exceção

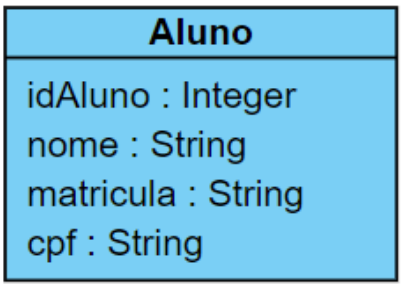
| **Resposta:** | |
|---|---|
| **Justificativa:** | |

**Prática:**

1. Crie um projeto do Java:
   Neste projeto, desenvolva uma entidade "Aluno", conforme abaixo:

   a. Utilize o padrão JavaBean

      i. Atributos privados
      ii. Sobrecarga de construtores
      iii. Métodos de encapsulamento set e get
      Sobrescrita do método toString()




2. Crie um banco de dados no MySql e desenvolva neste banco de dados uma tabela "Aluno" com os mesmos campos da classe de entidade.

3. Utilize o Pattern Factory Method para gerar a conexão com o banco de dados.

4. Crie um repositório "AlunoRepository" contendo métodos para inserir, alterar, excluir e consultar alunos na base de dados (utilize JDBC). Utilize o padrão de repositório genérico.

5. Através da Classe Program.java, demonstre a execução.