1. Defina:

    a. Interfaces
    b. Métodos abstratos
    c. Métodos estáticos
    d. Polimorfismo

2. Explique a função de cada palavra reservada em Java:

    a. `super`
    b. `static`
    c. `implements`
    d. `extends`
    e. `throws`

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

```
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:	

4. 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:	

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

```
interface A{
	public void print();
}

class B implements A{

}

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

- O código compila
- O código não compila

Resposta:	
Justificativa:	

6. 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:	

7. Baseado no SOLID, marque Verdadeiro ou Falso:

   a. ( ) Módulos de alto nível não devem depender da implementação de módulos de baixo nível, mas sim de suas abstrações.
   b. ( ) É uma boa prática criar interfaces especificas para cada módulo / camada de uma aplicação visando reduzir o acoplamento entre elas.
   c. ( ) Cada classe ou módulo de um sistema deve ser coeso em resolver apenas uma responsabilidade específica.
   d. ( ) Alto acoplamento é uma boa característica de projetos orientados a objetos