**Nachdenkzettel Interfaces**

**Aufgabe 1:**

Wenn man eine neue Methode in ein Interface einfügt, müssen alle implementierten Klassen, in diesem Fall die Klasse ‚B‘, die neue Methode implementieren, um weiterhin als gültige Implementierung des Interfaces zu gelten.

Die „Robustheit“ des Interfaces stellt also sicher, dass eine klaren „Vertrag“ zwischen dem Interface und den implementierten Klassen gibt. Wird das Interface aktualisiert, müssen die Klassen sicherstellen, dass sie diesem Vertrag entsprechen, indem sie die neuen Methoden implementieren. Dies fördern die Konsistenz und Zuverlässigkeit in der Programmierung.

**Aufgabe 2:**

Das Ganze kann problematisch werden, wenn man eine Applikation für verschiedene Branchen, Kunden oder Fälle entwickelt, da die Anwendung schwer anpassbar oder erweiterbar ist und zudem nicht besonders übersichtlich ist. Dies passiert insbesondere dann, wenn verschiedene Implementierungen von „X“ benötigt werden. Abhilfe kann man durch eine Schnittstelle schaffen, um die Erstellung von Objekten zu extrahieren. Die beste Möglichkeit dazu, bietet das Erstellen einer Factory.

Factories sind Entwurfsmuster und bieten eine Möglichkeit, diese Schnittstelle zu implementieren, indem sie die Erstellung von Objekten abstrahieren. Dadurch wird der Code flexibler und anpassungsfähiger, insbesondere wenn unterschiedliche Implementierungen von "X" benötigt werden.

**Aufgabe 3:**

Für außenstehende Methoden sind sowohl öffentliche Methoden als auch Konstanten ersichtlich. Das bedeutet, dass alle Methoden, die als Teil des Interfaces deklariert sind, für außenstehende Objekte sichtbar und aufrufbar sind. Wenn das Interface Konstanten enthält (public static final), sind diese ebenfalls sichtbar und können verwendet werden.

Die Implementierungsdetails sind jedoch für externe Objekte nicht sichtbar, da ein Interface nicht auf Implementierungsdetails eingeht.

**Aufgabe 4:**

Ja das ist möglich, wenn zwei Interfaces zufällig die gleiche Methode haben.