Kato, Thomas Reise 13.04.2024

Übung 1 – Software Engineering

Frage 1: Wie können Sie unter Berücksichtigung der Prinzipien des objektorientierten Entwurfs dafür sorgen, dass der Code, der den beiden Implementierungen gemeinsam ist, nicht dupliziert wird?

Antwort 1: Durch das Anlegen einer abstrakten Klasse, welche das Interface „NumberTransformer“ implementiert. Hier kann ich ungültige Werte abfangen und somit eine Fehlermeldung zurückgeben für alle Klassen, die von dieser abstrakten Klasse erben. Um die Transformation weiterhin berechnen zu können wird eine abstrakte Methode deklariert, welche in den Unterklassen entsprechend implementiert werden müssen. Diese Methode wird ausgeführt, falls die Eingabenummer gültig ist. (siehe Implementation der Aufgabe)

Frage 2: Wie kann die Objekterzeugung mit Hilfe einer zusätzlichen Klasse durchgeführt werden? In welchem Package sollte diese zusätzliche Klasse liegen?

Antwort 2: Es lässt sich eine Factory-Klasse implementieren, welche die entsprechende Instanz kreiert und zurückgibt. Sie sollte in einem eigenen Package sein, da die Client-Klasse nicht zu wissen braucht, von welchem spezifischen Typ die Instanz ist.

Frage 3: Welches Entwurfsmuster liegt für diesen Anwendungsfall nahe? Welchen Vorteil bringt die Nutzung dieses Entwurfsmusters?

Antwort 3: Das Fabrikmuster ermöglicht die Erzeugung eines Objektes mittels eines Methodenaufrufs anstelle des Konstruktors. Durch die Nutzung des Fabrikmusters wird eine einfache Erweiterbarkeit sichergestellt. Durch das Implementieren der Factory-Schnittstelle und eine Creator-Methode können neue Klassen eingeführt werden.

Frage 4: Warum sollten Testfälle in einer separaten Test-Klasse implementiert werden?

Antwort 4: Durch die Trennung des Test-Codes vom Produktionscode wird eine bessere Organisation geschaffen. Der Entwickler bzw. das Team weiß somit, welche Teile zur Produktion gehören und welche die Funktionalitäten testen sollen. Außerdem ermöglicht die Nutzung von Testframeworks eine einfache Ausführung aller Tests innerhalb eines Pakets. Außerdem möchte man in der Regel auch die Sichtbarkeit der Variablen bewahren, ohne sie wegen bestimmter Tests ändern zu müssen. Genauso muss die Sichtbarkeit getestet werden. Sprich Tests sollen so nah an der Realität wie möglich sein und dies ist nur möglich, wenn der Code von außen getestet wird.

Frage 5: Wozu dienen die Äquivalenzklassen im Blackbox-Test?

Antwort 5: Äquivalenzklassen unterteilen die möglichen Eingabewerte in zusammenhängende Gruppen, wobei jede Ä.klasse eine spezifische Gruppe von Eingabewerten repräsentiert. Dadurch ist eine effizientere Planung der erforderlichen

Tests möglich. Dabei sollen sowohl gültige als auch ungültige Benutzereingaben, sowie Randfälle, getestet werden.

Frage 6: Warum lässt sich für die Klasse Client nicht ohne weiteres ein Blackbox-Test umsetzen?

Antwort 6: Die Klasse Client lässt sich ohne weiteres nicht per Blackbox-Test testen, da sie auf die Implementierung der Methode „transformNumber“ zugreift. Je nach Instanz ist die Implementation anders. Klasse Client kann daher nicht isoliert getestet werden. Ein möglicher Test würde die Logik der Instanz einer Klasse, welche „NumberTransformer“ implementiert, umfassen.