Praktikum 2 AP2

* 1. Struktur konzeptionieren

Was ist die Oberklasse?

Oberklasse ist Produkt

Was sind die Unterklassen?

DiscountedProduct; 🡪 RefurbishedProduct; RefurbishedProduct ist eine Unterklasse des DiscountedProduct weil sie auch einen Discount als Parameter erbt, nur diese beiden Klassen haben diese Eigenschaft

LimitedProduct;

Welche Eigenschaften und Methoden gehören zur Oberklasse, weil diese allgemein sind?

Eigenschaften : name : String, basePrice : Double, margin : Int

Methoden: salesPrice(); toString(); addReview

Welche Eigenschaften und Methoden gehören in die Unterklassen, weil diese spezifisch sind?

Eigenschaften: discount: Int; quality : Quality (enum); totalNumberOfProducts: Int;

Methoden: moneySaved();

Welche Eigenschaften und Methoden müssten von welcher Unterklasse überschrieben / erweitert werden?

Alle Unterklassen: salesPrice(); toString(); addReview();

RefurbishedProduct überschreibt die Methode moneySaved() von der Klasse DiscountedProduct

2.1

Was ist dynamische Bindung? Geben Sie ein Beispiel.

Dynamic Binding löst die Bindung zur Laufzeit auf, da der Compiler nicht weiß, welche Implementierung zur Kompilierzeit verwendet werden soll.

Es wird auch als Late Binding bezeichnet, da die Bindung zu einem späteren Zeitpunkt zur Laufzeit und nicht zur Kompilierzeit aufgelöst wird. Dynamic Binding verwendet auch das Objekt zum Binden und nicht den Typ oder die Klasse; Das Überschreiben von Methoden ist ein Beispiel für die dynamische Bindung.

Quelle: https://www.delftstack.com/de/howto/java/static-and-dynamic-binding-in-java/

2.2 Typkompabilität

{val p1: Product = Product (" Buch ", 9.99 , 30)

val p2: Product = DiscountedProduct ("Süß igkeiten ", 1.99 , 30 , 20)

val p3: Product = RefurbishedProduct (" Handy ", 299.99 , 20 , 20 , Quality . USED )

VALIDE!!!

val p4: DiscountedProduct = p1 // p1 ist ein Produkt, weiß aber nichts von DiscountedProduct

Compilezeiterror

val p5: DiscountedProduct = p3 as RefurbishedProduct// alle Eigenschaften von DiscountedProduct vorhanden

val p6: DiscountedProduct = p2 as RefurbishedProduct // Hier fehlt dem DiscountedProduct die Quality

Laufzeitfehler /

val p7: RefurbishedProduct = DiscountedProduct ("Süß igkeiten ", 1.99 , 30 , 20)

val p8: Product = p1 as Any // Type ist any aber Produkt wird erwartet

val p9: Any = p1 // Any kann den Typ Product annehmen

3.1

Was ist ein Interface? Worin unterscheidet sich ein Interface zu einer Oberklasse?

4.2

Da Kindklassen von ihrer Elternklassen sämtlichen Eigenschaften erben besitzt die Subklasse automatisch sämtliche Methoden der Superklasse. Folglich würde die Subklasse nach dem Konzept der [**Vererbung**](http://www.inztitut.de/blog/glossar/vererbung/) ohne Methodendefiniton die Orginalmethode der Superklasse erben. Wenn wir die Methode in der Subklasse nun polymorph neu definieren, die Methode also überschreiben, ersetzen wir in Instanzen unserer Subklasse die allgemeine Methode der Superklasse durch eine spezielle Methodenfassung. Somit erhalten wir durch das Konzept der Polymorphe die Chance spezialisierte Funktionen in einer Kindklasse zu definieren.

Quelle: [Was ist Polymorphismus? Objektorientierte Programmierung verstehen! (inztitut.de)](https://www.inztitut.de/blog/glossar/polymorphismus/)