Übung 16: Lösung

- Richtlinien für die Verwendung der objektorientierten Konzepte in Hinblick auf Umsetzung der vorgestellten Entwurfsprinzipien
- Zu berücksichtigende objektorientierte Konzepte
  - Klasse (konkret/abstrakt)
  - Schnittstelle
  - Vererbung (Implementierungs- und Schnittstellenvererbung)
  - Assoziation
  - Schutzstufen (public, protected und private)
  - Polymorphie

Übung 16: Lösung

- Klasse und Schnittstellen allgemein
  - SoC (technische von fachlichen Belangen trennen etc.)
  - Konsistente und aussagekräftige Namensgebung
  - Anzahl Parameter in Methoden begrenzen
- Konkrete Klasse
  - Nicht für allgemeine Verwendung ausser Utility-Klassen
  - Statische Elemente mit Vorsicht
  - Schnittstelle und Implementierung trennen
  - Konstruktoren private oder protected oder package-protected (wenn möglich)
  - Wenn möglich „immutable“
  - Sichtbarkeit: package-protected (wenn möglich)

Übung 16: Lösung

- Abstrakte Klasse
  - Als Basisklasse für generische Teile
  - Default-Implementierungen von Schnittstellen anbieten
  - Keine leeren Default-Implementierungen
  - (Hilfs-)Methoden möglichst final machen
  - Sichtbarkeit: package-protected (wenn möglich)
- Schnittstelle
  - Wenn immer möglich gegen Schnittstellen programmieren
  - Sichtbarkeit: public oder package-protected (wenn möglich)
  - Als Abstraktion definieren
  - Schnittstellen-Segregation

Übung 16: Lösung

- Vererbung allgemein
  - Grosse Vererbungshierarchien vermeiden
  - „Ist ein(e)“ beachten
  - Nur „echte“ Gemeinsamkeiten von Unterklassen in die Basisklasse
  - Vorsicht vor Verwechslung mit Assoziation
- Implementierungsvererbung
  - Nur von abstrakten Klassen
  - Wenn von konkreten Klassen begründen und dokumentieren
  - Nach Möglichkeit durch Assoziation ersetzen
  - Mehrfachvererbung vermeiden/verbieten!

Übung 16: Lösung

- Schnittstellenvererbung
  - Konkrete oder abstrakte Klassen von Schnittstellen ableiten
  - Für verschiedene Clients verschiedene Schnittstellen anbieten (aber Kohäsion beachten!)
  - Gegen Schnittstellen programmieren
- Assoziation
  - Wenn immer möglich unidirektional
  - Auf Schnittstellen
  - Dependency Injection (DI)

Übung 16: Lösung

- Schutzstufen (public, protected und private)
  - Public nur für Methoden, die Clients benötigen (Schnittstelle)
  - Attribute immer private!
- Polymorphie
  - Höchstmögliche Abstraktionen (der Domäne) verwenden
  - Vor- und Nachbedingungen in den Implementierungen beachten
- Sonstiges
  - TellDon`tAsk bei Objektverwendung beachten
  - Ev. notwendige Abweichungen von diesem Leitfaden sind mit dem Architekten abzusprechen