Wo liegen die Unterschiede zwischen Qualitätsmanagement und Qualitätssicherung?
Qualitätssicherung (QS) besteht in der Durchführung von Verifikation und Validierung in jeder Phase der Software-Herstellung.

Qualitätsmanagement ist die Menge aller Aktivitäten, Vorgehensweisen, Techniken und Hilfsmittel, die sicherstellen, dass ein Software-Produkt vordefinierte Standards erreicht oder übertrifft.
Was ist ein Review?
„Ein Review ist ein [..] formal geplanter und strukturierter Analyse- und Bewertungsprozess, in dem Projektergebnisse einem Team von Gutachtern präsentiert und von diesen kommentiert oder genehmigt werden.“ [IEEE Std 610, Wallmüller 2001]
4 Arten von Test-Doubles nennen Dummy, Fake, Stub, Mock Welche Vorteile bringen Test-Doubles beim Testen im Bezug auf Qualitätssicherung? "

- Reduktion von Komplexität
- Reduktion von Abhängigkeiten zu anderen Systemteilen
- Isoliertes Testen einzelner Komponenten
- Indirekte Steuerung der zu testenden Komponente
- Kontrolle nicht deterministischer Werte (z.B. Datum/Zeit)
- Reduktion der Ausführungszeit

" Was ist ein Test Double? Generischer Begriff für den Austausch einer realen Komponente des Systems durch eine alternative, meist simplifizierte Implementierung für Testzwecke Welches sind die Phasen im Testprozess? "

- Planung & Steuerung (Vorbereitung)
- Analyse / Design (Spezifikation)
- Realisierung und Durchführung (Testläufe)
- Auswertung und Bericht (Testergebnisse)
- Abschluss (z.B. Archivierung).

" Was ist Software-Qualität? Die Beschaffenheit einer Einheit bezüglich ihrer Eignung festgelegte oder vorausgesetzte Erfordernisse erfüllen. Welche Möglichkeiten gibt es, Software-Qualität sicherzustellen? "
Konstruktive Qualitätsmaßnahmen:

- Technische Maßnahmen, z.B. Verwendung von Prinzipien, Methoden und Werkzeugen der Softwareentwicklung
- Organisatorische Maßnahmen, z.B. Verwendung von Vorgehensmodellen
- Menschliche Maßnahmen, z.B. Schulung der Projektmitarbeiter.

Analytische Qualitätsmaßnahmen: Prüfung und Bewertung der Qualität eines Produktes

- Statische Prüfungen, z.B. Reviews, Inspektionen, Audits
- Dynamische Prüfungen,z.B. Software Tests

" Was sind die Ziele von Refactoring? "
Steigerung der Code-Qualität, speziell Verbesserung der internen Struktur des Codes hinsichtlich

- Lesbarkeit
- Verständlichkeit
- Komplexität
- Architektur/Design

" Review von (Projekt-)Anforderungen "

- Reviews überprüfen die Korrektheit und Konsistenz von Artefakten an wohldefinierten Punkten des Entwicklungsprozesses.
  - Review der Anforderungen nach Änderungen gegen die Projektbeschreibung
  - Review später hergestellter Dokumente gegen die aktuell gültigen Anforderungen
  - Rückverfolgbarkeit der Anforderungen
- Anforderungen nach Kriterien einteilen
  - Funktional, nichtfunktional, testbar, verständlich
  - Erstrebenswert ist eine klare, knappe Beschreibung, welche umsetzbar ist
- Erstrebenswerte Kriterien aufzeichnen
  - Konsistenz, Vollständigkeit
  - Beispiel Review Issue:
    - ""Spezifikation des Fehlerfalls ist notwendig um der Projektbeschreibung zu genügen""

" Nennen Sie 3 Prinzipien des Qualitätsmanagements und erklären Sie diese.
(Laut Vorlesung sehr prüfungsrelevant)
"

- Konkrete operationalisierbare Qualitätsmerkmale.
- Produkt- und projektabhängige Qualitätsplanung.
- Unabhängigkeit bei Qualitätsprüfungen.
- Mehraugenkontrolle bei Qualitätsprüfungen.
- Frühzeitige Entdeckung und Behebung von Fehlern und Mängeln.
- Bewertungen der eingesetzten Qualitätsmaßnahmen.
- Organisation in Form von Qualitätsmanagementsystemen, z.B. ISO 9001, CMM(I), SPICE (ISO 15504) u.ä.
- Rückkopplung der Ergebnisse der Qualitätsprüfungen.
- Prinzip der ständigen Verbesserung von Produkten und Prozessen

" Welche Aufgaben hat Continuous Integration? "

- Kontinuierliches und automatisiertes Bauen der Anwendung
- Automatisierte Testausführung
- Automatisierte Berechnung von Metriken (z.B. Testabdeckung)
- Liefert einen aktuellen Status zur Qualität des Produkts

" Zwei Merkmale für Functionality Features, Fähigkeiten, Sicherheitsvorrichtungen Zwei Merkmale für Usability Menschliche Faktoren, Hilfe, Dokumentation Zwei Merkmale für Reliability Häufigkeit eines Ausfalls, Vorhersehbarkeit, Wiederherstellbarkeit Zwei Merkmale für Performance Antowotzeiten, Durchsatz, Genauigkeit Zwei Merkmale für Supportability Anpassungsfähigkeit, Wartung, Konfigurationen Ziele von (technischen) Reviews "

- Stärken und Schwächen desPrüfobjektes identifizieren
- (Entwicklungsprozess verbessern)

" Ziele von Inspektionen "

- Schwere Defekte im Prüfobjekt identifizieren
- Entwicklungsprozess verbessern
- Metriken ermitteln

" Ziele von Code-Walkthrough "

- Defekte und Probleme des Prüfobjektes identifizieren
- Ausbildung von Benutzern und Mitarbeitern

" Merkmale eines Black-Box-Tests? "

- Basiert auf Spezifikation.

- Wissen über innere Struktur wird ignoriert.
- Daten-getriebener Test.
- Input-Output-getriebene Tests.
- Anforderungsüberdeckung.
- Partitionierung (Äquivalenzklassen) der Eingabe-Daten

" Merkmale eines White-Box-Tests? "

- Basiert auf der Struktur von Codebzw. SE Modellen.
- Wissen über innere Struktur notwendig.
- Logik-getriebene Tests.
- Überdeckung von Knoten, Kanten, Pfaden.
- Partitionierung (Äquivalenzklassen) von Daten für Bedingungsauswertung

" Was ist ein Modultest? "Komponententest, Unittests

- Ziel: Aufspürung von Fehlern in der Implementierung.
- Module werden jeweils einzeln gegen ihre Spezifikation getestet: Übergabe von Daten, Prüfung des Outputs.
- Modultests erlauben eine genaue Lokalisierung und frühe Erkennung von Implementierungsfehlern.
- Modultests können bei größeren Systemen sehr aufwendig werden; daher ist bei größeren Tests besonderer Wert auf strukturiertes Vorgehen und Dokumentation des tatsächlichen Vorgehens zu legen.

" Was ist ein Integrationstest? Welche Strategien gibt es dafür? "

- Ziel: Test der Interaktion zwischen Modulen.
- Zusammenführung von Modulen zu größeren Strukturen (Häufig durch Entwickler).
- Integrationsstrategien: „Big-Bang“, Top-Down, Bottom-Up, Build-Integration.
- Build-Integration und Integrationstest sind –sofern möglich –zu bevorzugen.

" Was ist ein Regressionstest? Test nach Durchführung von Änderungen Was ist ein Systemtest? "

- Ziel: Test des Gesamtsystems gegen die Anforderungsanalyse bzw. den System-Entwurf.
- Validierung bzw. Verifikation.

" Was ist ein Akzeptanztest? "

- Ziel: Validierung des Systems gegen die Kundenanforderungen.
- Vom Kunden durchgeführter Systemtest, meist mit einer Submenge an Testfällen aus dem Systemtest.

" Welche Arten von Testüberdeckung gibt es?
•Anweisungsüberdeckung
•Zweigüberdeckungstest
•Pfadüberdeckungstest
•Bedingungsüberdeckungstest
Was ist Anweisungsüberdeckung? "
Statement Coverage

- Ziel: jede Anweisung muss mind. einmal durchlaufen werden.
- Vollständige Anweisungsüberdeckung liegt vor, wenn sämtliche Anweisungen mindestens einmal durchlaufen werden.
- Zeigt ob toter Code existiert
- Anweisungen, die niemals durchlaufen werden können
- Zu schwaches Kriterium für sinnvolle Testdurchführung

" Was ist das Ziel der Zweigüberdeckung? Was zeigt sie? Wobei hilft sie? "

- Ziel: Jede Kante muss mind. einmal durchlaufen werden
- Zeigt nicht ausführbare Programmzweige auf
- Hilft oft durchlaufene Programmteile gezielt zu optimieren

" Wie unterscheiden sich Zweig und Anweisungsüberdeckung? Im Gegensatz zum Anweisungsüberdeckungstest muss jede Entscheidung mindestens einmal true und false annehmen Was sind die Probleme der Zweigüberdeckung? "
Problematik:

- Unzureichender Test von Schleifen
- Komplexe Logik in Statements wird nicht berücksichtigt

" Grafik für Zweigüberdeckung "" Englisch für Pfadüberdeckung Path Coverage Ziel der Pfadüberdeckung Ziel: Abdeckung aller Pfade von Start- bis Endknoten Probleme der Pfadüberdeckung "

- Für komplexe Softwaremodule meist nicht durchführbar
- Unendlich hohe Anzahl von Pfaden
- Schleifen bieten extrem viele Pfade

" Grafik für Pfadüberdeckung "" Englisch für Bedingungsüberdeckung Condition Coverage Ziel der Bedingungsüberdeckung
Überprüfung zusammengesetzter Entscheidungen, Teilentscheidungen
Was ist ein einfacher Bedingungsüberdeckungstest? Fordert den Test aller atomaren Teilentscheidungen gegen Wahr und Falsch. Was ist ein mehrfacher Bedingungsüberdeckungstest?
Fordert den Test aller Wahrheitswertekombinationen der atomaren Teilentscheidungen (aufwendig)
Zwei Beispiele für einfachen Bedingungsüberdeckungstest
•Testfall 1: A = false, B = false, C = false, D = false
•Testfall 2: A = true, B = true, C = true, D = true
Wie viele Testfälle gibt es bei einem mehrfachen Bedingungsüberdeckungstest mit 4 Variablen? 2^4=16 Grafik für einfachen Bedingungsüberdeckungstest "" Grafik für Mehrfach-Bedingungsüberdeckungstest "" Welche Views gehören im V-Modell zu welchen Tests? "" Teilnehmer eines (technischen) Reviews
Moderator, Autor, Gutachter, Protokollführer
Charakteristika eines Reviews "

- Ausgebildeter Moderator
- Prüfteam gibt Empfehlung an Manager

" Ziele einer Inspektion "

- Schwere Defekte im Prüfobjekt identifizieren
- Entwicklungsprozess verbessern
- Metriken ermitteln

" Teilnehmer einer Inspektion
Moderator, Autor, Gutachter, Protokollführer, (Vorleser)
Charakteristika einer Inspektion "

- Ausgebildeter Moderator
- Prüfobjekt wird vom Vorleser Absatz für Absatz vorgetragen
- Moderator gibt Freigabe

" Ziele eines Code Walk-Through "

- Defekte und Probleme des Prüfobjektes identifizieren
- Ausbildung von Benutzern und Mitarbeitern

" Teilnehmer eines Code Walk-Through Autor (=Moderator), Gutachter Charakteristika eines Code Walk-Through "

- Prüfobjekt wird vom Autor ablauforientiert vorgetragen
- Autor entscheidet

" Grafik für Ablauf einer Review "" Welche Dinge müssen für ein Review geplant werden?
Objekt, Prüfziele, Auslösekriterien (Einstiegskriterien), Teilnehmer, Ort, Zeit.
Vorbesprechung einer Review
Vorstellung des Prüfobjekts bei komplexen und neuen Produkten
Durchführung einer Review
Gemeinsames Lesen.
Aufzeichnung von Mängeln.
Während des Reviews sollen Mängel entdeckt, nicht korrigiert werden
Nachbearbeitung einer Review
In der Nachbearbeitung werden dokumentierte Mängel korrigiert und in der Bewertung überprüft.
Typische Reviewdauer 2h Beispiele für Nicht-Ziele von Refactoring "

- Keine Änderung des sichtbaren Verhaltens
- Keine neue Funktionalität
- Keine Behebung von Fehlern
- Keine Änderungen, die nicht das Ziel haben, die Lesbarkeit bzw. Verständlichkeit des Codes zu verbessern, z.B. Performance Verbesserungen

" Welche Kategorien von Anforderungen gibts es? FURPS: Functional, Usability, Reliability, Performance, Supportability Was beschreiben funktionale Anforderungen? Welche Form haben sie?
Sie beschreiben **was** das System tut, um Mehrwert für die Stakeholder zu liefern.
Sie haben typischerweise folgende Form: Das System kann Funktion X ausführen.
Was sind nichtfunktionale Anforderungen? "

- ""Qualitätsanforderungen"" (wie „gut“)
- Erwünschte Charakteristika des Systems

" UML-Sichten auf Softwarestrukturen "

- Konzeptionelle Sicht
  Gibt eine Übersicht über den zu untersuchenden Problembereich, z.B. Domänenmodell.
- Spezifizierende Sicht
  Hier betrachten wir Software-Schnittstellen (keine Implementierungen, z.B. Komponentendiagramm).
- Implementierende Sicht
  Bei dieser Sichtweise beobachten wir Klassen und legen die Grundlage für die Implementierung. Dies ist vermutlich die am häufigsten eingenommene Betrachtungsweise, z.B. Klassendiagramm.

" What is crowdsourcing?
“The act of undertaking any external software engineering tasks by an undefined, potentially large group of online workers
in an open call format.” (Mao et al., 2016)
Welche Arten von Reviews mit Kunden gibt es? SRR, PDR, CDR, IPR Welche Reviews ohne Kunden gibts es? MR, Inspection, Code Walkthrough, Technical Review Was sind Inspektionen?
Inspektionen sind spezielle Reviews, die speziell in frühen Phasen der
Softwareentwicklung angewandt werden können.

Inspektionen sind formale, effiziente und wirtschaftliche Methoden um
Fehler im Design und Code zu finden.
Was ermöglichen Inspektionen?
Inspektionen ermöglichen eine systematische und strukturierte Fehlererkennung.
Welches ist die wichtigste Anwendung für Inspektionen?
Fehlerfindung in Anforderungs- und Designdokumenten
Welche Lesetechniken gibt es? Ad hoc, Checklistenbasiert, Szenariobasiert, Perspektivenbasiert, Fehlerbasiert Was kennzeichnet checklistenbasiertes Lesen? "

- Vordefinierte Fragestellungen die sequentiell auf das zu untersuchende Objekt angewandt werden.
- Die Checklisten sind auf zu findende Fehlertypen, Fehlerorte (Design, Code) usw. abgestimmt.
- In der Praxis meist generische Checklisten + Projektspezifische Erweiterungen.
- Einfache Anwendbarkeit durch Inspektoren.

" Perspektivenbasiertes Lesen "

- Fehlersuche aus verschiedenen Perspektiven: User, Tester, Entwickler.
- Zielsetzung: Unterschiedliche Fehler aus individuellen Sichten finden.
- Erfordert entsprechende Fachspezialisten.

" Welche Perspektiven gibt es beim perspektivenbasierten Lesen? User, Tester, Entwickler Usage-based Reading "Best-Practice Inspection

- Typischerweise existieren Szenarien (z.B. Use Cases)
- Priorisierung von Anwendungsfällen durch Fachleute.
- Zielsetzung: Fehler in wichtigen Anwendungsfällen zuerst finden.

" Ablauf eines Usage-Based Reading Reviews "

1. Auswahl des wichtigsten Anwendungsfalls
2. Überprüfung der relevanten Teile, die diesen Anwendungsfall beschreiben
3. Reporting von Mängeln und Fehlern
4. Auswahl des nächsten Anwendungsfalls

" Nutzen von Inspektionen "

- Einsparungen durch frühes Finden und Entfernen von Fehlern.
- Genauere Informationen über das Produkt und den Entwicklungsprozess zur Planung der Aktivitäten und Ressourcen (PM und QM).
- Beurteilungsgrundlage für die Produktqualität (z.B. durch Restfehlerschätzungen).
- Besseres Produktwissen der Inspektoren.
- Gemeinsame Sicht des Inspektionsteams auf das Produkt.
- Inspektion als Lerninstrument für neue Teammitglieder.

" Was ist Qualitätsmanagement? Qualitätsmanagement ist die Menge aller Aktivitäten, Vorgehensweisen, Techniken und Hilfsmittel, die sicherstellen, daß ein Software-Produkt vordefinierte Standards erreicht oder übertrifft. Woraus besteht Qualitätssicherung? Qualitätssicherung (QS) besteht in der Durchführung von Verifikation und Validierung in jeder Phase der Software- Herstellung. Wo im V-Modell sind Tests anzufinden? "" Welches sind die Schritte im TDD? Wie sieht das als Grafik aus? "

1. Think: Auswahl der zu implementierenden Anforderungen. Spezifikation der Testfälle.
2. Red: Implementierung und Ausführung der Testfälle. Alle Tests müssen fehlschlagen.
3. Green: Implementierung der Komponenten und Klassen und Ausführung der Testfälle. Testfall erfolgreich -> weiter bei Schritt 4. Testfall schlägt fehlt -> weiter bei Schritt 3.
4. Refactor:Änderung und Optimierung der Implementierung ohne Änderung der Funktionalität; Ausführung der Testfälle. Testfälle dürfen nicht fehlschlagen. Auswahl der nächsten Anforderung (Schritt 1)

" Ist jeder Testfall, jeder Fehler, jede Anforderung gleich viel „wert“? Nein. Beispiel: 10 unterschiedliche Bezahlverfahren einer Online-Plattform, wobei 80% des Umsatzes immer über dasselbe Verfahren abgewickelt wird. => Value-Based Testing

Requirements-Based testing => Welche Anforderungen bringen dem Kunden den meisten nutzen?
Risk-Based testing => Welche Risiken gefährden diesen Nutzen?
Test case selection techniques => Welche Testfälle addressieren dieses Risiko?
Welche Frage stellt die Verifikation? „Bauen wir das Produkt richtig?“

Umsetzung im Vergleich zur Spezifikation in vorangegangenen Phasen. Test der Umsetzung gegen die Spezifikation.
Welche Fragen stellt die Validierung? Was ist sie also? „Bauen wir das richtige Produkt?“

Entspricht die Lösung und damit die Spezifikation dem, was der Kunde erwartet?

Test der Umsetzung gegen die Nutzeranforderungen.
Wo sind Verifikation und Validierung im V-Modell? "" Was versteht man nach Myers unter Software Testen? „Testen ist das Ausführen eines Programms, mit der Absicht, Fehler zu finden.“ (Myers, 1979) Was ist ein Fehler laut Verifikation? „Ein Fehler ist eine Abweichung zwischen einem Programm und dessen Spezifikation“ (Kaner et al, 1999) Was ist ein Fehler laut Validierung? „Ein Software-Fehler liegt vor, wenn ein Programm nicht das tut, was ein Endbenutzer vernünftigerweise erwartet.“ (Myers, 1979) Wie unterscheidet sich Testen von Debugging? Testen beschäftigt sich nicht mit der Lokalisierung von Fehlerstellen und dessen Entfernung (vgl. Debugging) Kann durch Tests Fehlerfreiheit nachgewiesen werden? Nein (vgl. Formaler Korrektheitsbeweis). Welches sind die Phasen im PDCA-Zyklus? "

- Plan: Planen der nächsten Iteration des SPI-Zyklus basierende auf aktuellen Erfahrungswerten (inkl. Zielsetzung)
- Do: Durchführen der geplanten Aktivität (Task, Projekt usw)
- Check: Überprüfung der Produkt- und Prozessergebnisse (z.B. durch Messung)
- Act: Analyse der Ergebnisse als Feedback für den nächsten SPI Zyklus. Verbesserung der Erfolgsfaktoren, Vermeidung von Wiederholungsfehlern, Beseitigen von Schwachstellen usw.

SPI = Software Process Improvement
" Welche Ebenen gibts im PDCA-Zyklus? Arbeitspakete (work space, task level), Projektebene, Unternehmensebene (Management, Qualitätsmanagement) Was ist der Unterschied zwischen Qualitätsmanagement und Qualitätssicherung? Qualitätssicherung ist Verifikation und Validierung. Qualitätsmanagement eine größere Menge von Möglichkeiten um vordefinierte Standards zu erreichen. Wie werden Black Box Tests in agilen Umgebungen verwendet? Entwickler programmieren auf Grundlage der User-Stories und Abnahmekriterien. ZEITGLEICH entwickeln Tester Tests auf denselben Grundlagen. Dabei sind alle Black-Box Verfahren wie Äquivalenzklassenbildung, Grenzwertanalyse, Entscheidungstabellen und Zustandsbasierte Tests möglich. Ein Beispiel für verhaltensgetriebene Entwicklung (Behaviour Driven Development) "Given the ""Unregistered User has navigated to the ""register"" page

When entering ""newuser"" in the ""Username"" field
And entering ""abc"" in the ""Password"" field
And pressing the ""Register"" button

Then the text ""Thank you for Registering"" should appear on page
And the URL should end with ""use/accountPage""
" Welche Stufen von Akzeptanzkriterien gibt es? "" Beispiel für eine Bedingung bzw. Anforderung (Condition of Satisfaction) beim Login Passwörter dürfen nicht knackbar sein Beispielhafte Akzeptanzkriterien (Acceptance Criteria) für das Passwort in einer Registrierung. "Definition für ""nicht einfach knackbar"" aus Condition of Satisfaction:
Min 8. Zeichen, nur alphanumerisch, mind. eine Zahl, ...
" Wie sehen Beispiele für Testdaten bei Akzeptanzkriterien aus? "
" Teufelsquadrat nach Sneed "
Die Fläche des Quadrats ändert sich nicht.
Erhöhung der Qualität und Senkung der Entwicklungsdauer
funktioniert nur durch Reduktion des Umfangs und/oder Erhöhung der Entwicklungskosten
"
Integration von QS in RUP
"

- QS mit zugeordneten Rollen ist ständiger Begleiter während des gesamten Projektverlaufs.
- Review bei allen Meilensteinen.
- Eigene „Test Discipline“

" Wie ist QS im V-Modell XT integriert? "

- QS ist als eigenständiger Vorgehensbaustein realisiert.
- Dieser Vorgehensbaustein ist ein verpflichtender Bestandteil aller Projekttypen.
- Der Projekttyp „Einführung und Wartung eines organisationsspezifischen Vorgehensmodells“ beinhaltet den Gedanken der ständigen Verbesserung

" Wie ist QS in agilen Entwicklungsprozessen integriert? "

- Reviews und Tests als fixer Bestandteil der Sprints.
- Test-Driven Development

" Welche Typen von Messungen gibt es? "

- Direkte vs. indirekte Messungen:
  - Direkt
  - Indirekte
- Objektive vs. Subjektive Messungen
  - Objektiv
  - Subjektive
- Quantitative vs. Qualitative Daten:
  - Quantitativ
  - Qualitativ

" Was ist eine direkte Messung?
Wertermittlung direkt beim zu untersuchenden Objekt (z.B. Dauer, Aufwand einer Aufgabe)
Was ist indirekte Messung?
Ermittlung von Messwerten aus direkten Messungen (z.B. Effizienz einer Fehlererkennungsmethode = Anzahl der gefundenen Fehler pro Zeiteinheit)
Was kann objektiv gemessen werden? LoC, Auslieferungsdatum, Aufwand usw. Was wird subjektiv gemessen?
Messergebnisse basierend auf der individuellen Sichtweise des Betrachters, z.B. Fragebögen zur Erfassung der Kundenzufriedenheit.
Was sind quantitative Daten?
Daten als konkrete Zahlenwerte (z.B. für statistische Auswertungen)
Was sind qualitative Daten?
Visualisierte Informationen (Text, Bilder), z.B. durch Interviews, Interpretationen
Was sind Äquivalenzklassen? Zerlegung einer Menge von Daten (Input oder Output) in Untermengen (Klassen), die äquivalente Ergebnisse oder Auswirkungen produzieren. Welche Methoden zur Qualitätssicherung gibt es? "
Organisatorische Methoden

- Schaffen notwendige Infrastruktur
- Definieren und steuern den Qualitätsprozess

Dynamische Methoden

- Die Software bzw. Teile davon müssen ausführbar sein
- z.B. Testen

Statische Methoden

- Die Software muss nicht ausführbar bzw. integriert sein
- Fokus auf die interne Struktur

- z.B. toolgestützte Code Analysen
- z.B. Code Reviews

" Grafik zu Methoden der Qualitätssicherung "" Was ist ein Dummy-Objekt? "

- Platzhalter ohne Funktionalität
- Dient als „leerer“ Methoden Parameter
- Wird in die Komponente gereicht, jedoch nicht verwendet

" Was ist ein Fake-Objekt? "
Ausführbare Implementierung
Dient jedoch nicht zur Steuerung/Überprüfung des Testobjekts

Einsatzzwecke:

- Reale Implementierung zu langsam
- Reale Implementierung nicht verfügbar

Beispiele:

- Simulation einer Datenquelle
- In-Memory Datenbank

" Was ist ein Stub? "
Ausführbare Implementierung
Liefert vordefinierte Werte/Exceptions an den Aufrufer
Ermöglicht indirekt die Steuerung der zu testenden Komponente

- Steuerung, welcher Pfad in der zu testenden Komponente durchlaufen wird
- Erlaubt dadurch das Testen von Pfaden in der Komponente, die von außen nicht beeinflusst werden könnten

" Was ist ein Mock? "
Ausführbare Implementierung
Liefert vordefinierte Werte/Exceptions an den Aufrufer
Im Gegensatz zum Stub werden die an den Mock übergebenen Parameter überprüft und sind somit Teil des Tests
Ein Mock erkennt, wenn unerwartete Werte eingehen

- Direkte Verwendung von Assertions
- Auslösen von Exceptions

" Welches Arten von Tools zur Qualitätssicherung gibt es? Plus Beispiel "
Tools zur Überprüfung der Testabdeckung

- Clover
- Cobertura
- JaCoCo

Tools zur Überprüfung von Coding Standards

- Checkstyle
- IDE, z.B. IntelliJ

Tools zum Finden von Fehler Patterns

- FindBugs
- SonarLint

" Welche Ziele hat Testautomatisierung? "

- Wiederkehrende/repetitive manuelle Tätigkeiten reduzieren
- Hohe Anzahl an Tests in angemessener Zeit ausführen
- Tests durchführen, die man nicht manuell durchführen kann (z.B. Last-Tests)
- Automatisch Reports der Testergebnisse generieren
- Die Effizienz des Testens zu steigern

" Welche vier technischen Dimensionen gibt es bei Inspektionen? Was gehört jeweils dazu? "" Welches sind die Einflussfaktoren bei der Software-Entwicklung? Genau jene aus dem Teufelsquadrat!
Qualität, Quantität, Entwicklungsdauer, Kosten Welches sind die Schritte im Spiralmodell? "

1. Kick-off: Bestimme Ziele und Alternativen
2. Einschätzung der Risiken der Alternativen
3. Entwicklung und Test
4. Plan der nächsten Phase; weiter bei 1.

" Welche Leistungen bietet eine QS-Stelle für ein Projekt? "

- Qualitätsplanung als Teil der Projektplanung.
- Herstellen lokaler Standards auf Projekt- und Organisationsebene.
- Review zentraler Projektdokumente.
- Organisieren von Reviews: Ausbildung, Planung, Durchführung, Verbesserungsvorschläge.
- Unterstützung bei Personalauswahl und Software-Zukauf.
- Vorbereitung und Auswertung von Produkttests

" Wie ist ein Testfall aufgebaut? "
Ein Testfall besteht aus einer Menge (V, E, A, R)

- V: Vorbedingungen(z.B. System-oder DB-Zustände).
- E: Eingabewerte.
- A: Aktionenzur Eingabe der Werte.
- R: erwartete Ergebnisse

" Wann ist ein Test erfolgreich? Was soll er andernfalls anbieten?
Ein Test ist erfolgreich, wenn er einen Fehler gefunden hat.

Tests, die keinen Fehler aufdecken, sollen bessere Information über die Produktqualität liefern (systematische Testüberdeckung!).