- Free Quiz
- Software Testing
  - 1 Fundamentals of Testing
    - Why is Testing Necessary?
      - Human -> Error (mistake) -> Defect (fault, bug) which when executed may cause -> Failure
      - Measures the quality of the software
      - Gives confidence in the quality
      - Reduces the overall level of risk
      - How much testing? Depends on risk, safety & project constraints
    - Testing Objectives
      - Finding Defects
      - Providing information for decision-making
      - Preventing defects
      - Gaining confidence about the level of quality
    - Seven Testing Principles
      - Testing shows presence of defects
      - Exhaustive testing is impossible
      - Early testing
      - Defect clustering
      - Pesticide paradox
      - Testing is context dependent
      - Absence-of-error fallacy
    - Fundamental Test Process
      - Planning & Control
      - Analysis & Design
      - Implementation & Execution
      - Evaluating Exit Criteria & Reporting
      - Test Closure
    - The Psychology of Testing
      - Mindset of Developer & Tester
      - Communication in a constructive manner
      - Test Independence
    - Code of Ethics
      - Code is necessary, among other reasons, to ensure information accessed by testers are not put to inappropriate use
        - Public
        - Client and Employer
        - Product
        - Judgement
        - Management
        - Profession
        - Colleagues
        - Self
  - 2 Testing Throughout the Software Life Cycle
    - Software Development Models
      - Waterfall
      - Sequential
      - V-model
      - Iterative-Incremental
        - Agile
        - Scrum
    - Test Levels
      - Component (Unit)
      - Componet Integration
      - System
      - System Integration
      - Acceptance
    - Test Types
      - Black Box
        - Functional Testing
        - Non-Functional Testing (Software Characteristics)
      - White Box
        - Structural Testing
      - Testing Related to Change
        - Re-Testing
        - Regression
    - Maintenance Testing
      - Change to deployed software system or its enviroment
      - Triggered by
        - Modification
        - Migration
        - Retirement
      - Extensive regression testing required
  - 3 Static Techniques
    - Review Types
      - Informal Review
      - Walkthrough
      - Technical Review
      - Inspection
    - Activities of a Formal Review
      - Planning
      - Kick-off
      - Individual Preparation
      - Review Meeting
      - Rework
      - Follow-Up
    - Static Analysis by Tools
      - Find defects in software source code and models
      - Can locate defects that are hard to locate in dynamic testing
  - 4 Test Design Techniques
    - Test Development Process
      - Test Design Specification
      - Test Case Specification
      - Test Procedure Specification
    - Specification-based or Black-box Techniques
      - Equivalence Partitioning
      - Boundary Value Analysis
      - Decision Tables
      - State Transition Diagrams / Tables
      - Use Case Testing
    - Structure-base or White-box Technique
      - Statement Testing & Coverage (weakest)
      - Decision Testing & Coverage (stronger than Statement)
      - Other Structure-based Techniques
        - Condition Testing
        - Multiple Condition testing
        - All Path Testing (Strongest)
    - Experience-based Techniques
      - Exploratory Testing
      - Error Guessing
    - Choosing test Techniques
      - Risk & Objectives
      - Type of System & Dev Cycle
      - Regulatory Standards
      - Time & Budget
      - Knowledge & Experience
  - 5 Test Management
    - Test Organisation
      - Independent Testing
      - Tasks of Test Leader & Tester
    - Test Planning & Estimation
      - Test Planning
        - Test Policy, Strategy
        - Estimating Techniques
        - Test Plan
      - Test Approaches
        - Analytical
        - Model-based
        - Methodical
        - Process-Compliant or Standard-Compliant
        - Regression-Averse
        - Dynamic and Heuristic
        - Consultative
      - Entry & Exit Criteria
    - Test Progress Monitoring & Control
      - Test Monitoring
      - Test Reporting & Control
      - Test Summary Report
    - Risks and Testing
      - Risk: Probability / Likelihood & Impact
      - Project & Product Risks
      - Risk-based testing approach
    - Configuration Management
      - Establish and maintain the integrity of the products and ensure all items of test-ware are identified, version controlled, tracked for changes, related to each other.
    - Incident Management
      - Incident Management
      - Incident Logging
      - Test Incident Report
  - 6 Tool Support for Testing
    - Types of Test Tools
      - Management of Testing & Tests
      - Performance & Monitoring
      - Specific testing Needs
      - Test Execution & Logging
      - Test Specification
      - Static Testing
    - Effective Use of Tools
      - Potential Benefits & Risks
      - Special consideration for Test Execution, Static Analysis & Test Management tools
    - Introducing a Tool into an Organisation
      - Main Considerations
      - Start with a Pilot project
      - Success factors for deployment

Software Testing

http://www.rogeriodasilva.com/

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs.
It involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:
meets the requirements that guided its design and development,
responds correctly to all kinds of inputs,
performs its functions within an acceptable time,
is sufficiently usable,
can be installed and run in its intended environments, and
achieves the general result its stakeholders desire.
As the number of possible tests for even simple software components is practically infinite, all software testing uses some strategy to select tests that are feasible for the available time and resources.

Visit: http://www.rogeriodasilva.com/

Fundamentals of Testing

Testing is a process rather than a single activity. This process starts from test planning then designing test cases, preparing for execution and evaluating status till the test closure. So, we can divide the activities within the fundamental test process into the following basic steps: 1) Planning and Control 2) Analysis and Design 3) Implementation and Execution 4) Evaluating exit criteria and Reporting 5) Test Closure activities

Test Design Techniques

Following are the typical design techniques in software engineering: 1. Deriving test cases directly from a requirement specification or black box test design technique. The Techniques include: Boundary Value Analysis (BVA) Equivalence Partitioning (EP) Decision Table Testing State Transition Diagrams Use Case Testing 2. Deriving test cases directly from the structure of a component or system: Statement Coverage Branch Coverage Path Coverage LCSAJ Testing 3. Deriving test cases based on tester's experience on similar systems or testers intuition: Error Guessing Exploratory Testing

Tool Support for Testing

http://www.rogeriodasilva.com/10-best-free-defect-tracking-tools/

The Classification Tree Method is a method for test design, as it is used in different areas of software development. It has been developed by Grimm and Grochtmann in 1993. Classification Trees in terms of the Classification Tree Method must not be confused with decision trees.
The classification tree method consists of two major steps:
Identification of test relevant aspects and their corresponding values as well as
Combination of different classes from all classifications into test cases.
The identification of test relevant aspects usually follows the specification of the system under test. These aspects form the input and output data space of the test object.
The second step of test design then follows the principles of combinatorial test design.
While the method can be applied using a pen and a paper, the usual way involves the usage of the Classification Tree Editor, a software tool implementing the classification tree method.

See: 10 Best Free Defect Tracking Tool
http://www.rogeriodasilva.com/10-best-free-defect-tracking-tools/

Test Management

- Test Procedure - Manual
- Test Script - Automated

Test management most commonly refers to the activity of managing the computer software testing process. A test management tool is software used to manage tests (automated or manual) that have been previously specified by a test procedure. It is often associated with automation software.

Testing Throughout the Software Life Cycle

There are various software development approaches defined and designed which are used/employed during development process of software, these approaches are also referred as “Software Development Process Models” (e.g. Waterfall model, incremental model, V-model, iterative model, etc.). Each process model follows a particular life cycle in order to ensure success in process of software development. Software life cycle models describe phases of the software cycle and the order in which those phases are executed. Each phase produces deliverables required by the next phase in the life cycle. Requirements are translated into design. Code is produced according to the design which is called development phase. After coding and development the testing verifies the deliverable of the implementation phase against requirements. There are following six phases in every Software development life cycle model: Requirement gathering and analysis Design Implementation or coding Testing Deployment Maintenance

Static Techniques

In integrated circuit design, dynamic logic is a design methodology in combinatory logic circuits, particularly those implemented in MOS technology. It is distinguished from the so-called static logic by exploiting temporary storage of information in stray and gate capacitances. It was popular in the 1970s and has seen a recent resurgence in the design of high speed digital electronics, particularly computer CPUs. Dynamic logic circuits are usually faster than static counterparts, and require less surface area, but are more difficult to design. Dynamic logic has a higher toggle rate than static logic but the capacitative loads being toggled are smaller so the overall power consumption of dynamic logic may be higher or lower depending on various tradeoffs. When referring to a particular logic family, the dynamic adjective usually suffices to distinguish the design methodology, e.g. dynamic CMOS or dynamic SOI design.
Dynamic logic is distinguished from so-called static logic in that dynamic logic uses a clock signal in its implementation of combinational logic circuits. The usual use of a clock signal is to synchronize transitions in sequential logic circuits.

Free Quiz

http://www.rogeriodasilva.com/istqb-iseb-certified-tester-foundation-level/

Take the free Quiz. http://www.rogeriodasilva.com/istqb-iseb-certified-tester-foundation-level/