It's a Final Year..

Rules in Class

1. **ATTENDANCE** will be monitored strictly
2. If you are present - **PAY ATTENTION** and **PARTICIPATE**
3. **NO PLAGARISM** & **NO COPYING**. Any plagiarism detected will result in immediate grade reduction.
4. Any **PROBLEM**, contact **ME** soonest possible
5. **ALERT** on the **DEADLINE** of all assignments' submission, quizzes, tests and final assessment.

Overview of Software Quality

DR JULIANA JAAFAR

Learning Objective

To introduce the concept of:-

- Software Quality
- Software Quality Attributes/Characteristics/Factors
- Software Quality Management
- Software Quality Assurance (SQA)

Quality

Quality

*"The **degree of excellence** of something.."* *(Cambridge Dictionary)*

Luxury

Meet Requirements

Taste

Fitness for use

Class/Standard

Software Quality

What is **Software (Software Product)**?

Software is *computer programs*, *procedures* and possibility associated *documentation* and *data* pertaining to the *operation of a computer system*

(IEEE)

Computer programs – the 'code'

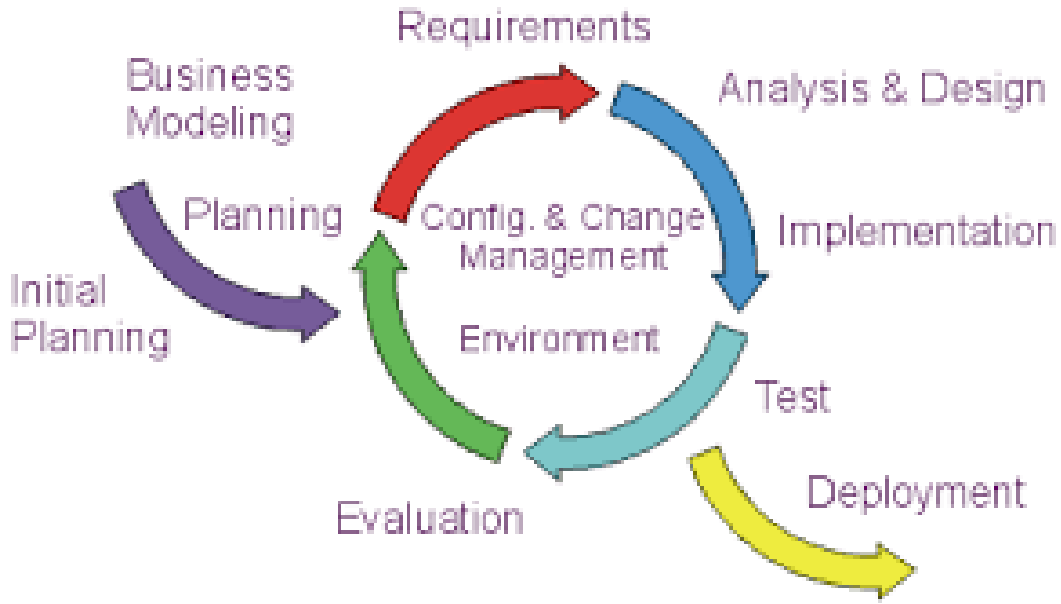
Procedures – order and schedule in which the programs are performed, the method employed, and the person responsible for performing the activities necessary for applying the software

Documentation – development documentation (e.g. SRS, SRD, various UML diagrams and etc.), user manuals, maintenance documents and etc.

Data – includes parameters, codes, name lists of the individual for operating the software, standard test data, etc.

What is **Software Process**?

A **software process** (also known as software methodology) is a set of related activities that leads to the production of the software.



Sub activities : requirement validation, architectural design, unit testing & etc.

Supporting activities: configuration and change management, quality assurance, project management, user experience.

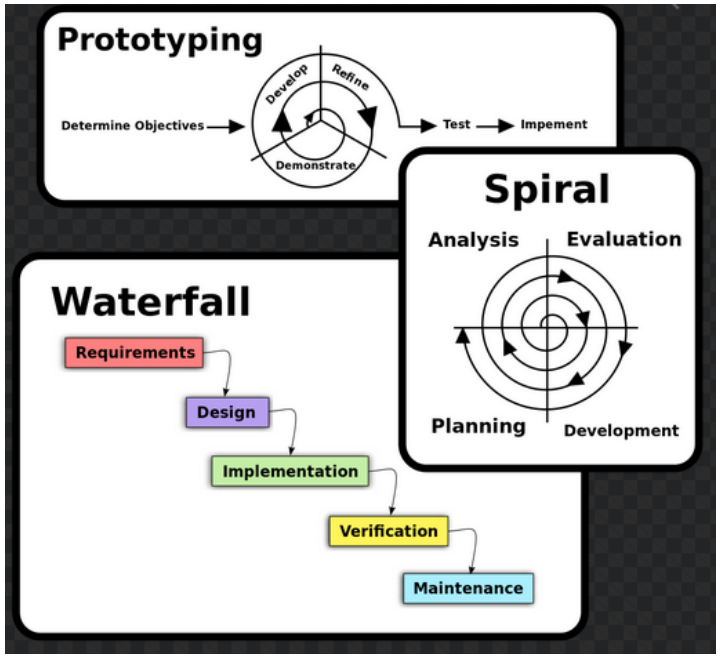
Other activities (process improvement): configuration and change management, quality assurance, project management, and user experience.

Software Process

COMPLEX

NO IDEAL software process. Its depends on the project, organization, resources, budget and timeline.

Example:
An organization works on critical systems has a very structured process, while with business systems, with rapidly changing requirements, a less formal, flexible process is likely to be more effective



Software Process Description

- **Products**: The outcomes of an activity.
  - Example: the outcome of architectural design may be a model for the software architecture.
- **Roles**: The responsibilities of the people involved in the process.
  - Example: the project manager, programmer, etc.
- **Pre and post-conditions**: The conditions that must be true before and after an activity.
  - Example: the pre-condition of the architectural design is the requirements that have been approved by the customer, while the post-condition is the diagrams describing the architecture have been reviewed.

Movie Time..

Software Disasters

- **July 28, 1962 -- Mariner I space probe.**
  A bug in the flight software for the Mariner 1 causes the rocket to divert from its intended path on launch. Mission control destroys the rocket over the Atlantic Ocean. The investigation into the accident discovers that a formula written on paper in pencil was improperly transcribed into computer code, causing the computer to miscalculate the rocket's trajectory.

- **World War III Almost (1983)**
  Cost: Nearly all of humanity
  Disaster: The Soviet early warning system falsely indicated the United States had launched five ballistic missiles. Fortunately the Soviet duty officer had a funny feeling in my gut and reasoned if the U.S. was really attacking they would launch more than five missiles, so he reported the apparent attack as a false alarm.
  Cause: A bug in the Soviet software failed to filter out false missile detections caused by sunlight reflecting off cloud-tops.

Reference: http://www.devtopics.com/20-famous-software-disasters

Software Disasters

- **Disastrous Study (1999)**
  Cost: Scientific credibility
  Disaster: In this ironic case, software used to analyze disasters had a disaster of its own. The New England Journal of Medicine reported increased suicide rates after severe natural disasters. Unfortunately, these results proved to be incorrect.
  Cause: A programming error caused the number of suicides for one year to be doubled, which was enough to throw off the entire study.

- **Y2K (1999)**
  Cost: $500 billion
  Disaster: One man s disaster is another man s fortune, as demonstrated by the infamous Y2K bug. Businesses spent billions on programmers to fix a glitch in legacy software. While no significant computer failures occurred, preparation for the Y2K bug had a significant cost and time impact on all industries that use computer technology.
  Cause: To save computer storage space, legacy software often stored the year for dates as two digit numbers, such as 99" for 1999. The software also interpreted 00" to mean 1900 rather than 2000, so when the year 2000 came along, bugs would result.

Reference: http://www.devtopics.com/20-famous-software-disasters

Software Disasters

- **Love Virus (2000)**
  Cost: $8.75 billion, millions of computers infected, significant data loss
  Disaster: The LoveLetter worm infected millions of computers and caused more damage than any other computer virus in history. The worm deleted files, changed home pages and messed with the Registry.
  Cause: LoveLetter infected users via e-mail, Internet chat and shared file systems. The email had an executable file attachment and subject line, ILOVEYOU. When the user opened the attachment, the virus would infect the user s computer and send itself to everyone in the address book.

  Cancer Treatment to Die For (2000)
  Cost: Eight people dead, 20 critically injured
  Disaster: Radiation therapy software by Multidata Systems International miscalculated the proper dosage, exposing patients to harmful and in some cases fatal levels of radiation. The physicians, who were legally required to double-check the software s calculations, were indicted for murder.
  Cause: The software calculated radiation dosage based on the order in which data was entered, sometimes delivering a double dose of radiation.

Reference: http://www.devtopics.com/20-famous-software-disasters

What can you learn from software disaster examples?

How can we prevent a similar situation from occurring in the future?

Why Software Quality Important?

Impact on our daily life C

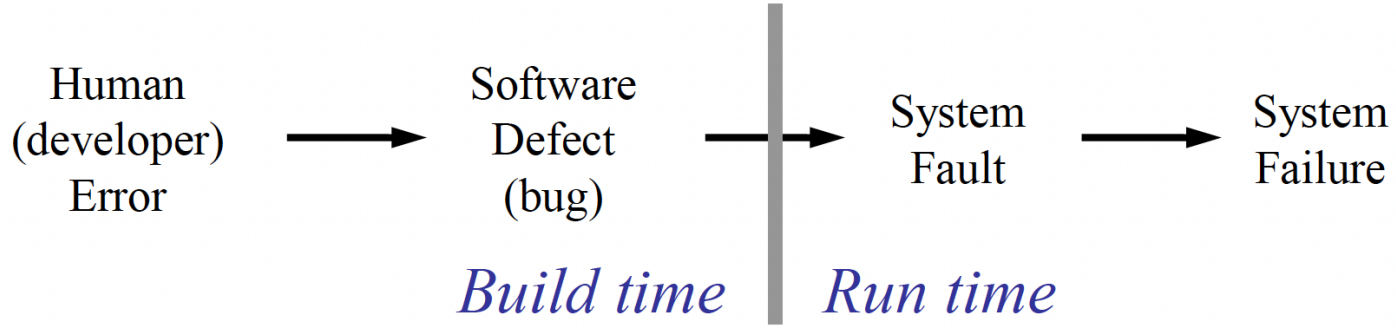
Computer system is in our **EVERYDAY** life.

Banking, Communication – social media, email, phone,
Education,
Healthcare, Transportation –automobile, aeronautics

COSTLY

- Money
- Reputation
- Loss of life

Error – Defect - Failure



Error

- **Definition:** Human action that produces an incorrect result [IEEE 610].
- **Example:**
  A programmer commits an error by re-using a piece of software which is not intended in the context of the current project (the Ariane 5 satellite launching rocket)

Defect

- **Definition:** A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g., an incorrect statement or data definition
- **Example:**
  After the programmer has re-used the piece of code in the wrong context, the software contains a defect/fault.

Failure

- **Definition:** Deviation of the component or system from its expected delivery, service or result [Fenton].
- **Example:**
  The Ariane 5 rocket crashes on its first mission. Cost ~ $7 billion

Causes of Software Errors

- Bad requirements
- Communication failures.
  - Client Vs Project Team
  - Among project team
- Improper design of software
- Coding errors/ Code complexity
- Non-compliance with documentation and coding instructions.
- Inadequate software testing prior to release
- Poor development methodologies
- Lack of top management support and commitment
- Reusing software/code that may already contain bugs
- Time pressures
- Changing technologies
- User interface and procedure errors.
- Documentation errors.

Is Software with **Less Defects** or **Less Failures** is A High-Quality Software?

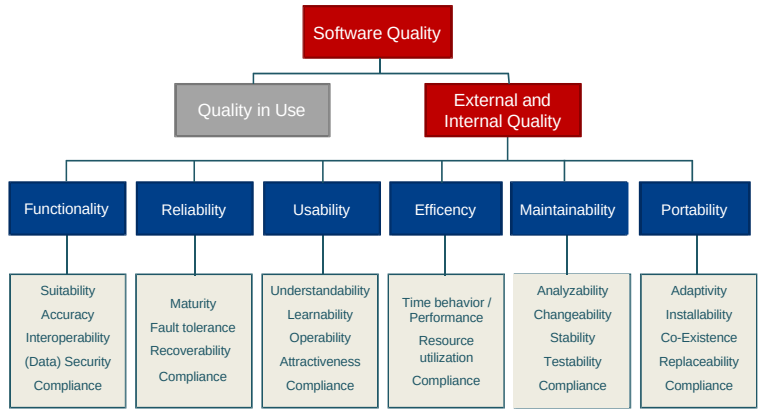
Software Quality Factors/Attributes

Product

- Correctness
- Robustness
- Security
- Ease of use
- Ease of learning
- Efficiency
- Extendibility
- Reusability
- Portability

Process

- Timeliness
- Cost-effectiveness
- Self-improvement



Source:
ISO/IEC 9126: Evaluation of software products, quality characteristics and guidance on usage

Software Quality Definition

Software quality attributes/characteristics/ factors

Software Quality Model

Quality comprises characteristics and significant features of a product or an activity which relate to satisfying of given requirements

(German Industry Standard DIN 55350 Part 11)

Quality is the totality of features and characteristics of a product or a service that bears on its ability to satisfy the given needs

(The ANSI/ASQC A3/1978 Standard)

The degree to which a system, component, or process meets specified requirements, and customer or user needs or expectations

(The IEEE Std 729-1983)

Conformance to explicitly stated functional and performance requirements explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software

(The Pressman's)

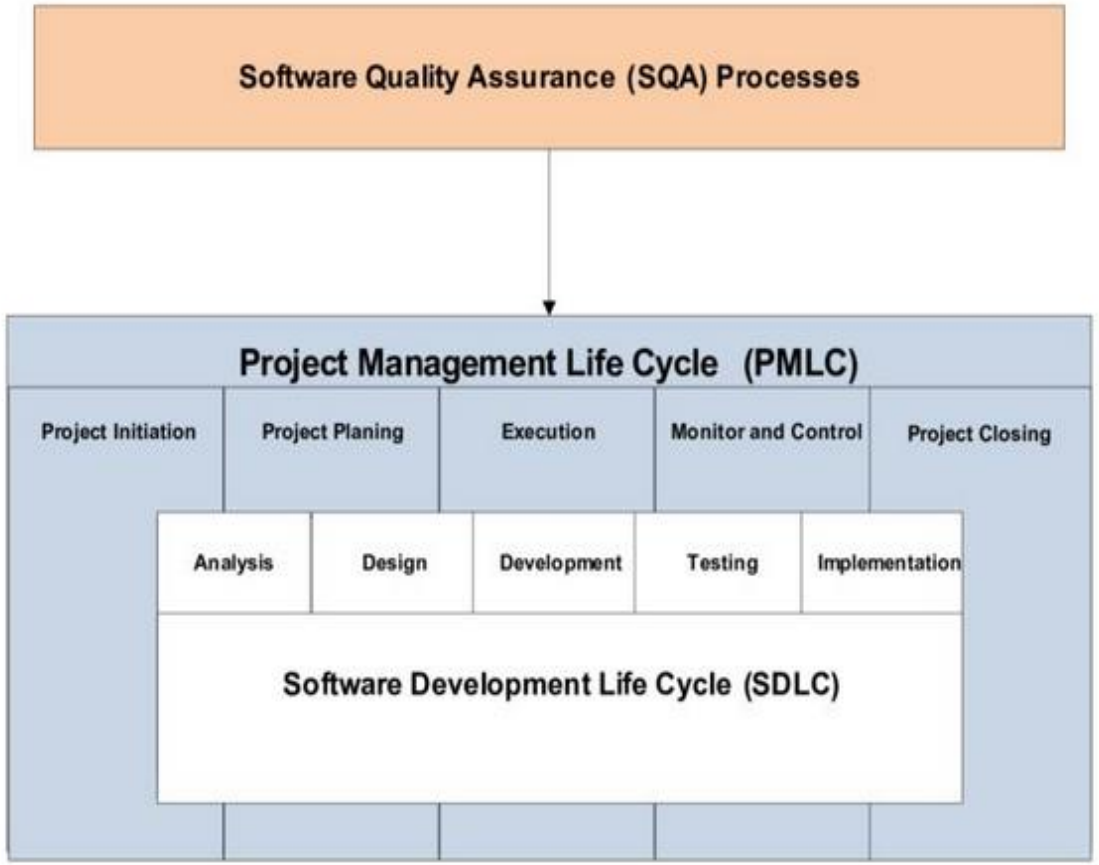
Software Quality

The combination of all **software components** is required to ensure the **quality** of the **development process** and the subsequent long years **of product maintenance**.

SOFTWARE QUALITY CHALLENGES

- **Defining it**
- **Describing it (qualitatively)**
- **Measuring it (quantitatively)**
- **Achieving it (technically)**

Software Quality Management (SQM)

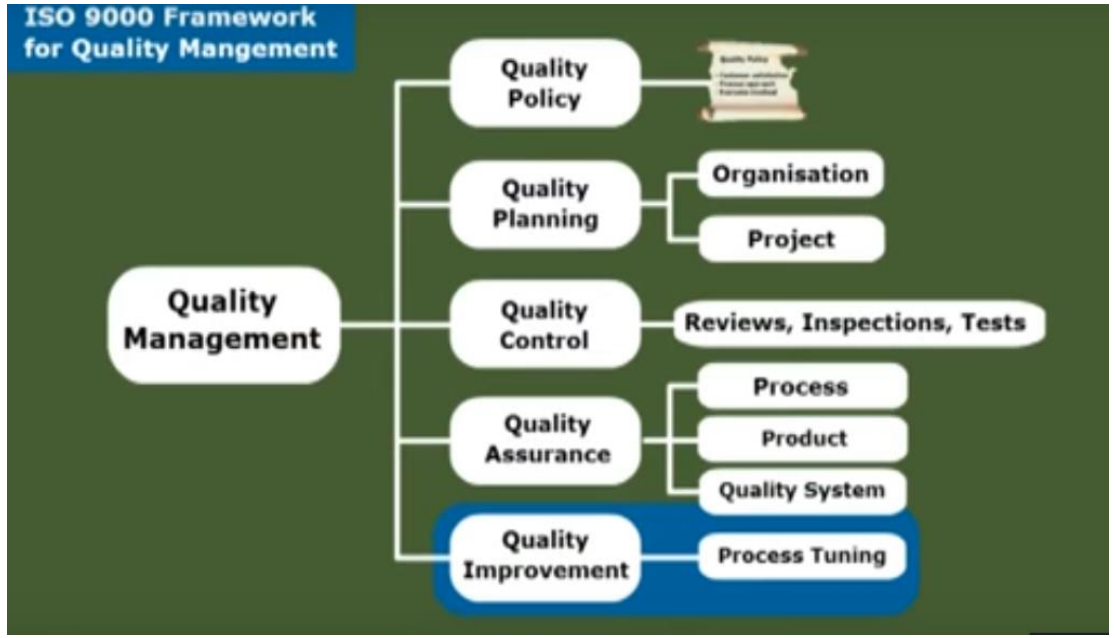


SQM - Is the process of managing the process of building a quality software

The activities involves:-

- **Plan quality management**: identifying the quality requirements and standards for the project and product (Quality Policy and Quality Planning)
- **Perform quality assurance:** auditing the quality requirements and quality control results to ensure that appropriate quality standards are used (Quality Assurance)
- **Control quality:** monitoring and recording the results of quality activities to assess performance and recommend necessary changes (Quality Monitoring and Control).
- **Improve Process:** Improve by measuring the quality of our work products and optimizing the development processes to eliminate the defects.

Software Quality Management Framework (ISO 9000)



ISO 9000 SQM Principles

- Know **who your customers** are
- Understand what they **value**
- Have a **mechanism to deliver that value at the right price**

SQA Plan

Quality management planning **determines the quality standards** that are applicable to **the project** and devising a way to satisfy them.

The goal is to create quality management plan which documents the following:

- The way the team will implement the quality policy/standards/practices
- The way the quality of both the project and the product will be assured during the project
- The resources required to ensure quality
- the additional activities necessary to carry out the quality plan

Software Quality Assurance

- A set of policies and activities to:
  - **Define** quality objectives (software product and software process)
  - Help **ensure** that software products and processes meet these objectives (i.e. reviews, auditing, inspections, testing)
  - **Assess** to what extent those quality objectives has achieved and complied (measure and reporting)
  - **Improve** them over time

SQA Objectives

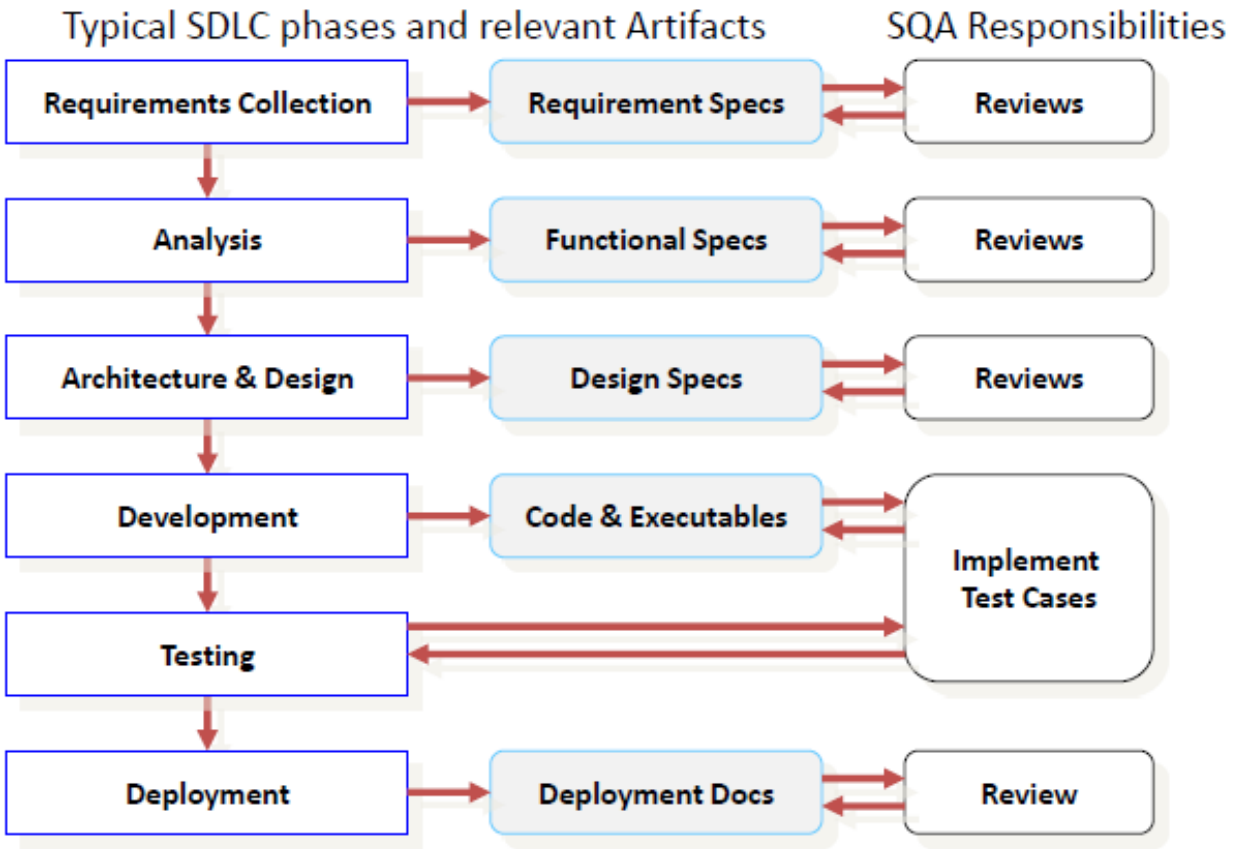
- To **improve software quality** by appropriately **monitoring** the **product** i.e. software, and the **development process** of the product
- To ensure full compliance with the **established standards** for the product and the process
- To **ensure any inadequacies** in the product, process or standards are brought to management attention, so these inadequacies can be fixed

SQA Roles in a Software Project

To ensure…

- An appropriate development methodology is in place
- The projects use standards and procedures in their work
- Independent reviews and audit are conducted
- Documentation is produced to support maintenance and enhancement
- The documentation is produced during and not after development
- Mechanism are in place and used to control changes
- Testing emphasizes all the high risks product areas
- Each software task is satisfactorily completed before the succeeding one begun
- Deviations from standards and procedures are exposed as soon as possible
- **The project is auditable by external professionals
- The quality control work is itself performed against established standards
- SQA plan and software development plan are compatible

SQA in SDLC


Typical SDLC phases and relevant Artifacts
SQA Responsibilities
Requirements Collection
Requirement Specs
Reviews
Analysis
Functional Specs
Reviews
Architecture & Design
Design Specs
Reviews
Development
Code & Executables
Implement Test Cases
Testing
Deployment
Deployment Docs
Review

SQA Principles

- **Practical quality features**
  - Observe the customer and **user quality requirements** from the beginning
  - Concrete quality features and if possible quantifiable
  - **Different project** and **different staff** can have **different features**
- **Product and Project-Dependent Quality Planning**
  - Consider the requirements and **life-time of software**
  - Project uncertainties and **risk** must also be taken into account
  - Comprehensive definition of requirements i.e. cover all attributes of software aspects and use of software such as reusability, usability, and maintainability
- **Checking of Results of Quality Tests**
  - A part of quality control activity
  - Discrepancies from the planned quality level be revealed
  - Peer **reviews**, **walkthrough**, **inspection** and **audits** are the techniques

SQA Principles

- **Multiple Quality Reviews/Audit/Walkthrough activities** - *"Not because people are untrustworthy, but because they are human"*
  - Do in group i.e. joint analysis of a document
  - Avoid one man show
- **Maximum Constructive Quality Assurance**
  - To **minimize errors/faults/bugs** in the **development** process and to minimize number or errors/faults/bugs
  - Identify the suitable preventive measures
- **Early Discovery and Correction of Errors and Faults**
  - The strategy must be to **recognize** and **eliminate** an **error as early as possible**

SQA Principles

- **Integrated Quality Assurance**
  - **Quality assurance** + entire **development process**
  - Put quality assurance measures in a proper plan and organize with other development measures
  - This principle makes the level of quality visible at each point in time
- **Evaluation of Applied Quality Assurance Measures**
  - Check the implementation of quality assurance organization and its measures
  - Call for **internal** and **external audits** (do the data collection, analysis, report, exam etc.)
  - Quality audits result will shows company current status of tools, techniques etc.
  - **Improve future strategy and plan** for software project and business

Challenges of SQA

- Software Production Vs Manufacturing Production
  - **Product Complexity**
  - **Product Visibility**
  - **Product Development Process**

VS

See you NEXT ⇨ Class..