UNIT I – FUNDAMENTALS

Principles of Testing – Phases of Software Project – Quality Assurance and Control – Verification and Validation - White Box Testing: Static Testing – Structural Testing – Challenges.

1.3 Quality Assurance and Control

Quality Assurance is a process-oriented approach that focuses on preventing defects during software development. It ensures that proper methods, standards, and procedures are followed throughout the Software Development Life Cycle (SDLC). QA aims to improve the development process and reduce the chances of errors before the software is delivered.

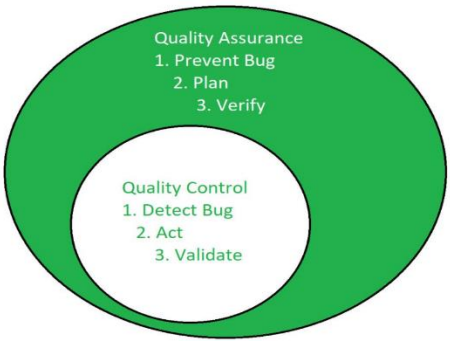
What is Quality Assurance (QA)?

Quality Assurance is defined as an activity that ensures the approaches, techniques, methods, and processes designed for the projects are implemented correctly. It recognizes defects in the process. Quality Assurance is completed before Quality Control.

- It focuses on preventing defects.
- It is a proactive process and is preventive in nature.
- It helps to recognize flaws in the process.
- These activities monitor and verify that the processes used to manage and create deliverables have been followed.

Characteristics of Quality Assurance (QA)

- **Process-Focused:**
  QA mainly concentrates on establishing and following effective processes and standards to maintain consistent software quality.
- **Preventive Approach:**
  Quality Assurance aims to avoid defects by improving development practices and ensuring quality is built into the software from the beginning.
- **Ongoing Improvement:**
  QA encourages regular evaluation and enhancement of processes to achieve better software quality over time.



Quality Control (QC):

Quality Control is a product-oriented approach that focuses on identifying and correcting defects after software development. It involves testing and inspecting the software to detect bugs, errors, and issues. QC ensures that the final software product meets the required quality standards and functions correctly before release.

What is Quality Control?

Quality Control (QC) is a software engineering process used to ensure that the software product meets the required quality standards. It verifies whether the methods, techniques, and procedures followed during development produce the expected results.

- QC mainly focuses on checking the quality of the final software product rather than the development process used to create it.
- It is a reactive process because it identifies and detects defects after development.
- Quality Control activities monitor and test project deliverables to confirm that they satisfy the specified quality requirements.

Characteristics of Quality Control (QC)

- **Reactive Approach:**
  QC focuses on identifying and correcting defects in the completed software product after development.
- **Defect Detection:**
  Through testing and inspection, QC aims to discover errors and issues as early as possible.
- **Product-Focused:**
  Quality Control involves examining and evaluating the software product to ensure it meets the required quality standards.

No	Quality Assurance	Quality Control
1	It is a procedure that focuses on providing assurance that quality requested will be achieved.	Quality Control is a procedure that focuses on fulfilling the quality requested.
2	In order to meet the customer requirements, QA defines standards and methodologies	QC confirms that the standards are followed while working on the product
3	QA is process oriented.	QC is product oriented.
4	QA aims to prevent the defect	QC aims to identify and fix defects
5	It's a Proactive measure	It's a Reactive measure
6	It is performed before Quality Control	It is performed only after QA activity is done
7	It does not involve executing the program	It always involves executing a program

No	Quality Assurance	Quality Control
8	QA involves in full software development life cycle	QC involves in full software testing life cycle
9	All team members are responsible for QA.	Testing team is responsible for QC.
10	It is the procedure to create the deliverables	It is the procedure to verify that deliverables
11	Less time-consuming activity	More time-consuming activity
12	QA ensures that everything is executed in the right way, and that is why it falls under verification activity	QC ensures that whatever we have done is as per the requirement, and that is why it falls under validation activity

1.4 Verification and Validation

Role of V & V

Verification and Validation (V&V) are essential processes in the Software Development Life Cycle (SDLC) that ensure software quality and correctness. Verification checks whether the software is developed according to the specified requirements and design, while validation ensures that the final product satisfies user needs and expectations.

- V&V improves the quality, reliability, and performance of software.
- Verification focuses on reviewing documents, design, and development activities, whereas validation focuses on testing the final software product.
- These processes help identify defects at an early stage, reducing development time and overall cost.

Validation

- **Validation** is the method for determining if software meets stated criteria during or at the conclusion of the development process.
- Validation is the method of determining if the completed product fulfills the customer's expectations and criteria. It is a dynamic validation and testing technique for the real product.

Example: Testing a shopping application where users verify login, product search, cart, and payment features work correctly.

Verification

Verification is the process of assessing software to evaluate if the results of a certain development phase meet the requirements established at the beginning of that phase.

Software Verification Testing is the process of examining documentation, designs, code, and programs to determine whether or not the software was constructed in accordance with the requirements. Verification operations include reviews, walk-throughs, and inspections.

Although verification can assist to identify whether the program is of good quality, it cannot guarantee that the system is functional. The purpose of verification is to determine if the system is well-engineered and error-free.

Example: Reviewing requirement and design documents of a banking app to ensure all features are correctly planned before development.

What is the difference between verification and validation?

Verification	***Validation***
1. Document, design, code, and program **verification** is a static practice.	1. **Validation** is a dynamic process used to validate and test the real product.
2. It does not include running the code.	2. It always entails running the code.
3. It is based on human **verification** of papers and data.	3. It is computer-based program execution.
4. **Verification** procedures include inspections, reviews, walk-through, and desk-checking, among others.	4. **Validation** employs techniques such as black box (functional) testing, gray box testing, and white box (structural) testing, among others.
5. **Verification** is the process of determining if the software meets standards.	5. **Validation** is the process of determining if software satisfies the expectations and needs of the client.
6. It can detect mistakes that **validation** cannot. It is a low-level activity.	6. It can detect flaws that **verification** cannot. It is a high-level activity.
7. The goal is to provide requirements, application and software architecture, high level, full design, and database design, among other things.	7. The goal is to create a real product: a unit, a module, a set of integrated modules, and a functional finished product.
8. The QA team completes the **verification** of the program to ensure that it meets the standards in the SRS document.	8. The testing team is involved in the **validation** process.
9. It is usually completed before **validation**.	9. It usually happens after **verification**.