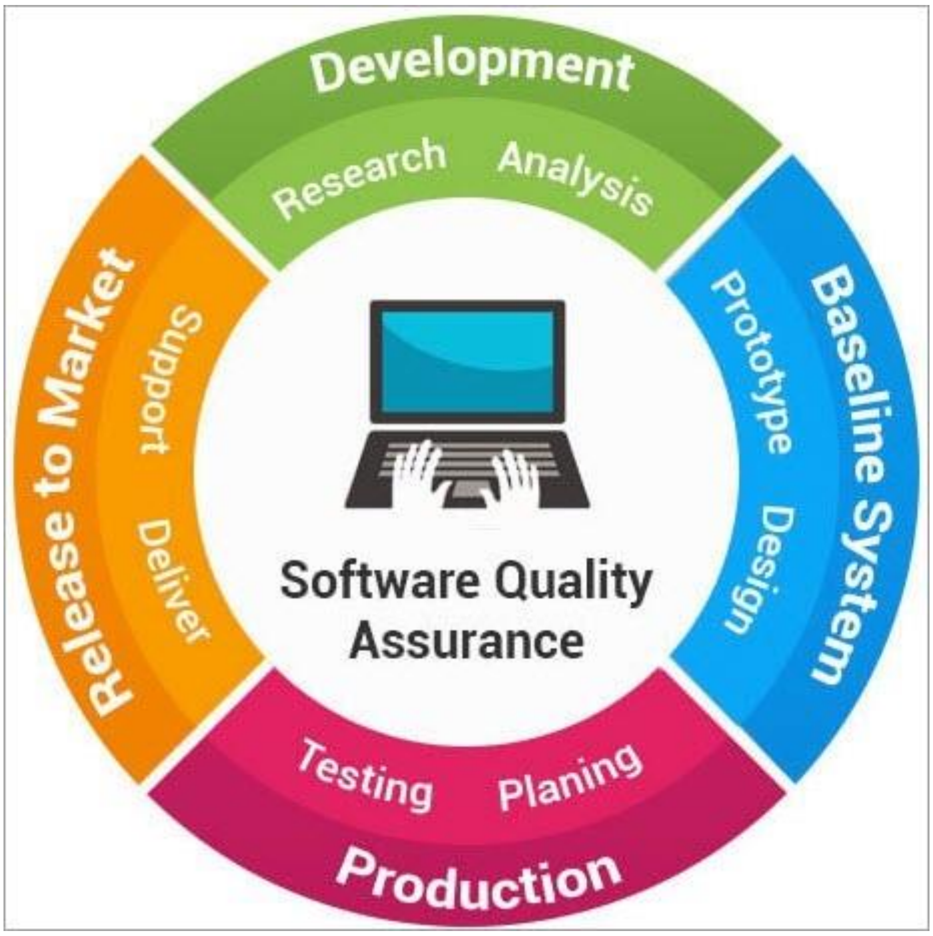
**SQA Plan:** plan to make sure that software follows requirements (SRS)

The Software Quality Assurance Plan comprises the procedures, techniques, and tools that are employed to make sure that a product or service aligns with the requirements defined in the SRS(Software Requirement Specification).



The plan identifies the SQA responsibilities of the team and lists the areas that need to be reviewed and audited. It also identifies the SQA work products.

**The SQA plan document consists of the following sections:**

1. Purpose
2. Reference
3. Software configuration management
4. Problem reporting and corrective action
5. Tools, technologies, and methodologies
6. Code control
7. Records: Collection, maintenance, and retention
8. Testing methodology

**SQA Activities**
**Given below is the list of SQA activities:**
**#1) Creating an SQA Management Plan**
Creating an SQA Management plan involves charting out a blueprint of how SQA will be carried out in the project with respect to the engineering activities while ensuring that you corral the right talent/team.

**#2) Setting the Checkpoints**
The SQA team sets up periodic quality checkpoints to ensure that product development is on track and shaping up as expected.

**#3) Support/Participate in the Software Engineering team's requirement gathering**
Participate in the software engineering process to gather high-quality specifications. For gathering information, a designer may use techniques such as interviews and FAST (Functional Analysis System Technique).

Based on the information gathered, the software architects can prepare the project estimation using techniques such as WBS (Work Breakdown Structure), SLOC (Source Line of Codes), and FP(Functional Point) estimation.

**#4) Conduct Formal Technical Reviews**
An FTR is traditionally used to evaluate the quality and design of the prototype. In this process, a meeting is conducted with the technical staff to discuss the quality requirements of the software and the design quality of the prototype. This activity helps in detecting errors in the early phase of SDLC and reduces rework effort later.

**#5) Formulate a Multi-Testing Strategy**
The multi-testing strategy employs different types of testing so that the software product can be tested well from all angles to ensure better quality.

**#6) Enforcing Process Adherence**
This activity involves coming up with processes and getting cross-functional teams to buy in on adhering to set-up systems.

**This activity is a blend of two sub-activities:**
- **Process Evaluation:** This ensures that the set standards for the project are followed correctly. Periodically, the process is evaluated to make sure it is working as intended and if any adjustments need to be made.
- **Process Monitoring:** Process-related metrics are collected in this step at a designated time interval and interpreted to understand if the process is maturing as we expect it to.

**#7) Controlling Change**
This step is essential to ensure that the changes we make are controlled and informed. Several manual and automated tools are employed to make this happen.

By validating the change requests, evaluating the nature of change, and controlling the change effect, it is ensured that the software quality is maintained during the development and maintenance phases.

**#8) Measure Change Impact**
The QA team actively participates in determining the impact of changes that are brought about by defect fixing or infrastructure changes, etc. This step has to consider the entire system and business processes to ensure there are no unexpected side effects.

For this purpose, we use software quality metrics that allow managers and developers to observe the activities and proposed changes from the beginning till the end of SDLC and initiate corrective action wherever required.

**#9) Performing SQA Audits**
The SQA audit inspects the actual SDLC process followed vs. the established guidelines that were proposed. This is to validate the correctness of the planning and strategic process vs. the actual results. This activity could also expose any non-compliance issues.

**#10) Maintaining Records and Reports**
It is crucial to keep the necessary documentation related to SQA and share the required SQA information with the stakeholders. Test results, audit results, review reports, change request documentation, etc. should be kept current for analysis and historical reference.

**#11) Manage Good Relations**
The strength of the QA team lies in its ability to maintain harmony with various cross-functional teams. QA vs. developer conflicts should be kept at a minimum and we should look at everyone working towards the common goal of a quality product. No one is superior or inferior to each other- we are all a team.