Testing Concepts 1

Pulkit Verma
Trainee

1. **Retesting** is a process to check specific test cases that are found with bug/s in the final execution. Generally, testers find these bugs while testing the software application and assign it to the developers to fix it. Then the developers fix the bug/s and assign it back to the testers for verification. This continuous process is called Retesting.

   Regression Testing is a type of software testing executed to check whether a code change has not unfavourably disturbed current features & functions of an Application

- Regression testing is performed for passed test cases while Retesting is done only for failed test cases.
- Regression testing checks for unexpected side-effects while Re-testing makes sure that the original fault has been corrected.
- Regression Testing doesn't include defect verification whereas Re-testing includes defect verification.
- Regression testing is known as generic testing whereas Re-testing is planned testing.
- Regression Testing is possible with the use of automation whereas Re-testing is not possible with automation.

2. Which of the one are part of functional testing -

   UAT, Integration, Regression

3. While developing a software or application product, it is tested at the final stage as a whole by combining all the product modules and this is called System Testing.
4. Confirmation testing is done only for new changes which is again unlike regression testing which is done for already existing features.
5. **Static Testing:**

   Static Testing is a type of a Software Testing method which is performed to check the defects in software without actually executing the code of the software application.

Static testing is performed in the early stage of development to avoid errors as it is easier to find sources of failures and it can be fixed easily. The errors that can't not be found using Dynamic Testing, can be easily found by Static Testing.

Dynamic Testing:

Dynamic Testing is a type of Software Testing which is performed to analyse the dynamic behaviour of the code. It includes the testing of the software for the input values and output values that are analysed.

6.

SDLC	STLC
SDLC is mainly related to software development.	STLC is mainly related to software testing.
Besides development other phases like testing is also included.	It focuses only on testing the software.
SDLC involves a total of six phases or steps.	STLC involves only five phases or steps.
In SDLC, more members (developers) are required for the whole process.	In STLC, less number of members (testers) are needed.
In SDLC, the development team makes the plans and designs.	In STLC, the testing team makes the plans and designs.
Goal of SDLC is to complete successful development of software.	Goal of STLC is to complete successful testing of software.
It helps in developing good quality software.	It helps in making the software defects free
SDLC phases are completed before the STLC phases.	STLC phases are performed after SDLC phases.

7. **Waterfall Model - Advantages**

- Simple and easy to understand and use

- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

Waterfall Model - Disadvantages

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

8. Functional testing is a type of testing that seeks to establish whether each application feature works as per the software requirements. Each function is compared to the corresponding requirement to ascertain whether its output is consistent with the end user's expectations. The testing is done by providing sample inputs, capturing resulting outputs, and verifying that actual outputs are the same as expected outputs.

   Some functional testing examples are:

- Can users successfully log in to the application once they provide legitimate credentials?
- Does the payment gateway reject the input and display an error message when a user keys in an invalid credit card number?

- Do inputs to the “Add New Record” screen successfully add and save a new record to the database?

9. No .The system as a whole can be tested only if all modules are integrated and all modules work correctly System testing should be done before UAT (User Acceptance testing) and Before Unit Testing.
10.

Verification	Validation
It includes checking documents, design, codes and programs.	It includes testing and validating the actual product.
Verification is the static testing.	Validation is dynamic testing.
It does *not* include the execution of the code.	It includes the execution of the code.
Methods used in verification are reviews, walkthroughs, inspections and desk-checking.	Methods used in validation are Black Box Testing, White Box Testing and non-functional testing.
It checks whether the software conforms to specifications or not.	It checks whether the software meets the requirements and expectations of a customer or not.

It can find the bugs in the early stage of the development.	It can only find the bugs that could not be found by the verification process.
The goal of verification is application and software architecture and specification.	The goal of validation is an actual product.
Quality assurance team does verification.	Validation is executed on software code with the help of the testing team.
It comes before validation.	It comes after verification.
It consists of checking documents/files and is performed by humans.	It consists of execution of a program and is performed by computer.

11.

- Stubs and Drivers are designed as a dummy for the missing or inaccessible modules or the components.
- Most commonly, stubs and drivers are used in the incremental integration testing, where stubs are used in top-bottom methodology while drivers are used in a bottom-up methodology.

STUBS	DRIVER
A section of code that imitates the called function is known as Stubs.	A section of code that imitates the calling function is known as Drivers.

It is used to test the functionality of modules and test modules and also replicate the performance of the lower-level module which are not yet merged, and the activity of the missing module/components.	When the main module is prepared or ready, we will take the help of drivers. Generally, drivers are a bit more complex as compared to the stubs.
The stubs are developed during the Top-down approach of incremental integration testing.	The drivers are developed during the bottom-up approach of incremental integration testing.
Stubs replicate the activity of not developed and missing modules or components.	Drivers authorise test cases to another system and which refer to the modules under testing.
The stubs are created by the team of test engineers.	Mostly, the drivers are created by the developer and the unit Test engineers.
Stubs are developed when high-level modules are tested, and lower-level modules are not formed.	Drivers are acquired when lower-level modules are tested, and higher-level modules are not yet developed.
These are parallel to the modules of the software, which are under development.	On the other hand, the drivers are used to reminding the component, which needs to be tested.
The stubs signify the low-level modules.	The drivers signify the high-level modules.
Fundamentally, the Stubs are also known as a called program and initially used in the Top-down integration testing.	The Drivers are also known as the calling program and are mainly used in bottom-up integration testing.
These are reserved for testing the features and functionality of the modules.	The drivers are used if the core module of the software isn't established for testing.

12. STLC is a very important phase of SDLC and the final product or the software cannot be released without passing through the STLC process.
13. Testing should start as early as possible in software cycle
14. False
15. True, regression tests to prove that the rest of the system has not been affected by the maintenance work.
16. Unit Testing is typically performed by the developer. In SDLC or V Model, Unit testing is the first level of testing done before integration testing. Unit testing is such type of testing technique that is usually performed by the developers

17. V Model is a highly disciplined SDLC model in which there is a testing phase parallel to each development phase. The V model is an extension of the waterfall model in which **testing is done on each stage parallel with development in a sequential way.**
18. Static testing include – Review, inspection, Walkthrough
19. True. Acceptance testing is most often focused on a validation type testing
20. True, Integration testing primarily focuses on verifying data communication among different modules of the software project.