Program Testing and Debugging

Introduction

- Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do.
- The benefits of testing include preventing bugs, reducing development costs and improving performance.
- Good testing is one that has the probability of finding errors in the program.

Cont.

Debugging:

- Debugging is the process of fixing a bug in the software.
- It can defined as the identifying, analyzing and removing errors.
- This activity begins after the software fails to execute properly and concludes by solving the problem and successfully testing the software.
- It is considered to be an extremely complex and tedious task because errors need to be resolved at all stages of debugging.

Differences between Testing and Debugging

Testing	Debugging
Testing is the process to find bugs and errors.	Debugging is the process to correct the bugs found during testing.
It is the process to identify the failure of implemented code.	It is the process to give the absolution to code failure.
Testing is the display of errors.	Debugging is a deductive process.
There is no need of design knowledge in the testing process.	Debugging can’t be done without proper design knowledge.
It is based on different testing levels i.e. unit testing, integration testing, system testing etc.	Debugging is based on different types of bugs.
Testing is a stage of software development life cycle (SDLC).	Debugging is not an aspect of software development life cycle, it occurs as a consequence of testing.
Testing is composed of validation and verification of software.	While debugging process seeks to match symptom with cause, by that it leads to the error correction.

Principles of Testing

- All the test should meet the customer requirements
- To make our software testing should be performed by a third party
- Exhaustive testing is not possible. As we need the optimal amount of testing based on the risk assessment of the application.
- All the test to be conducted should be planned before implementing it
- It follows the Pareto rule(80/20 rule) which states that 80% of errors come from 20% of program components.
- Start testing with small parts and extend it to large parts

Types of Testing.

Regression testing:

- Checking whether new features break or degrade functionality.
- Every time a new feature is added leads to changes in the program. This type of testing makes sure that the whole component works properly even after adding components to the complete program.

Smoke Testing

- Smoke testing, also called *build verification testing* or *build acceptance testing*, is none exhaustive software analysis that ascertains that the most crucial functions of a program work but does not delve into finer details.
- Smoke testing is the preliminary check of the software after a build and before a release. This type of testing finds basic and critical issues in an application before critical testing is implemented.

Cont.

Usability testing:

- Validating how well a customer can use a system or web application to complete a task.

Stress testing:

- Testing how much strain the system can take before it fails. Considered to be a type of non-functional testing.

Functional testing:

- Checking functions by emulating business scenarios, based on functional requirements. Black-box testing is a common way to verify functions.

Cont.

Performance testing:

- Testing how the software performs under different workloads. Load testing, for example, is used to evaluate performance under real-life load conditions.

Security Testing

- is a type of Software Testing that uncovers vulnerabilities of the system and determines that the data and resources of the system are protected from possible intruders.
- It ensures that the software system and application are free from any threats or risks that can cause a loss

Levels of Testing:

Unit Testing:

- It focuses on the smallest unit of software design. In this, we test an individual unit or group of interrelated units. It is often done by the programmer by using sample input and observing its corresponding outputs.

Integration Testing

- It is testing if there are errors that have arise as a result of combining/integrating components
- The aim is ensuring that software components or functions operate together.

Levels of testing: Cont.

Acceptance testing:

- Verifying whether the whole system works as intended as per the end-user specificatons.

Alpha Testing

- This is a type of validation testing. It is a type of *acceptance testing* which is done before the product is released to customers. It is typically done by Quality assurance people.

Beta Testing

- The beta test is conducted at one or more customer sites by the end-user of the software. This version is released for a limited number of users for testing in a real-time environment

Levels of testing: Cont.

System Testing

- This software is tested such that it works fine for the different operating systems.
- It is covered under the black box testing technique. In this, we just focus on the required input and output without focusing on internal working.
- In this, we have security testing, recovery testing, stress testing, and performance testing

Software Testing Best Practices

- **Continuous testing:** Project teams test each build as it becomes available.
  - This type of software testing relies on test automation that is integrated with the deployment process.
  - It enables software to be validated in realistic test environments earlier in the process – improving design and reducing risks.
- **Configuration management:** Organizations centrally maintain test assets and track what software builds to test.
  - Teams gain access to assets such as code, requirements, design documents, models, test scripts and test results.
  - Good systems include user authentication and audit trails to help teams meet compliance requirements with minimal administrative effort.

Cont.

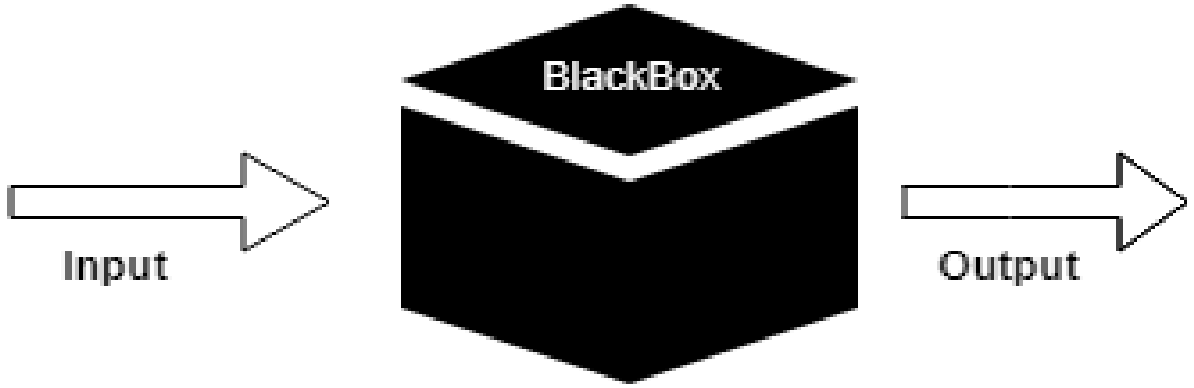
- **Metrics and reporting:** Reporting and analytics enable team members to share status, goals and test results.
  - Advanced tools integrate project metrics and present results in a dashboard.
  - Teams quickly see the overall health of a project and can monitor relationships between test, development and other project elements.
- **Defect or bug tracking:** Monitoring defects is important to both testing and development teams for measuring and improving quality.
  - Automated tools allow teams to track defects, measure their scope and impact, and uncover related issues.

Methods of Software Testing: Black Box testing

- Black box testing is a technique of software testing which examines the functionality of software without peering into its internal structure or coding.
- The primary source of black box testing is a specification of requirements that is stated by the customer.
- In this method, tester selects a function and gives input value to examine its functionality, and checks whether the function is giving expected output or not.
- If the function produces correct output, then it is passed in testing, otherwise failed.

Cont.

- The test team reports the result to the development team and then tests the next function.
- After completing testing of all functions if there are severe problems, then it is given back to the development team for correction.
- Black box testing is as per the diagram below



Techniques used in Blackbox testing

- **Decision Table Technique**: it is a systematic approach where various input combinations and their respective system behavior are captured in a tabular form. It is appropriate for the functions that have a logical relationship between two and more than two inputs.
- **Boundary Value Technique:** it is used to test boundary values, boundary values are those that contain the upper and lower limit of a variable. It tests, while entering boundary value whether the software is producing correct output or not.

Cont.

- **State Transition Technique:** it is used to capture the behavior of the software application when different input values are given to the same function. This applies to those types of applications that provide the specific number of attempts to access the application.
- **Cause-Effect Technique:** it underlines the relationship between a given result and all the factors affecting the result. It is based on a collection of requirements.
- **Use case Technique:** it is used to identify the test cases from the beginning to the end of the system as per the usage of the system.

White Box Testing

- Also known as glass box is **testing, structural testing, clear box testing, open box testing and transparent box testing**.
- It tests internal coding and infrastructure of a software focus on checking of predefined inputs against expected and desired outputs.
- It is based on inner workings of an application and revolves around internal structure testing. In this type of testing programming skills are required to design test cases.
- The primary goal of white box testing is to focus on the flow of inputs and outputs through the software and strengthening the security of the software.

Cont.

- The term 'white box' is used because of the internal perspective of the system. The clear box or white box or transparent box name denote the ability to see through the software's outer shell into its inner workings.
- Developers do white box testing. In this, the developer will test every line of the code of the program.
- The developers perform the White-box testing and then send the application or the software to the testing team, where they will perform the black box testing and verify the application along with the requirements and identify the bugs and sends it to the developer.

Activities involved in white box testing include

- **Path testing:** involves writing the flow graphs and test all independent paths. Here writing the flow graph implies that flow graphs are representing the flow of the program and also show how every program is added with one another.
- **Loop testing:** involves testing the loops such as while, for, and do-while, etc. and also check for ending condition if working correctly and if the size of the conditions is enough.
- **Condition testing:** This involves testing all logical conditions for both **true** and **false** values; that is, we will verify for both **if** and **else** condition.

Reasons for White Box Testing

- It identifies internal security holes.
- To check the way of input inside the code.
- Check the functionality of conditional loops.
- To test function, object, and statement at an individual level.

Advantages of White Box Testing

- White box testing optimizes code so hidden errors can be identified.
- Test cases of white box testing can be easily automated.
- This testing is more thorough than other testing approaches as it covers all code paths.
- It can be started in the SDLC phase even without GUI.

Disadvantages of White Box Testing

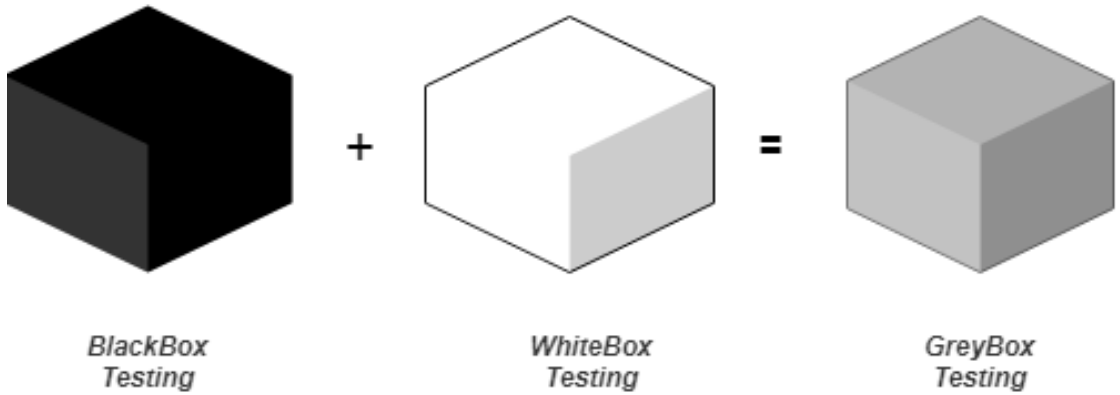
- White box testing is too much time consuming when it comes to large-scale programming applications.
- White box testing is much expensive and complex.
- White box testing needs professional programmers who have a detailed knowledge and understanding of programming language and implementation.

Difference between White Box and Black Box

White-box testing	Black box testing
The developers can perform white box testing.	The test engineers perform the black box testing.
To perform WBT, we should have an understanding of the programming languages.	To perform BBT, there is no need to have an understanding of the programming languages.
In this, we will look into the source code and test the logic of the code.	In this, we will verify the functionality of the application based on the requirement specification.
In this, the developer should know about the internal design of the code.	In this, there is no need to know about the internal design of the code.

Grey Box Testing

- Grey box testing is a software testing method to test the software application with partial knowledge of the internal working structure.
- It is a **combination of black box and white box testing** because it involves access to internal coding to design test cases as white box testing and testing practices are done at functionality level as black box testing.



Cont.

- Grey Box testing commonly identifies context-specific errors that belong to web systems.
- For example; while testing, if tester encounters any defect then he makes changes in code to resolve the defect and then test it again in real time.
- It concentrates on all the layers of any complex software system to increase testing coverage.
- It gives the ability to test both presentation layer as well as internal coding structure. It is primarily used in integration testing and penetration testing.

Why Grey Box testing?

- It provides combined benefits of both Blackbox testing and White Box testing.
- It includes the input values of both developers and testers at the same time to improve the overall quality of the product.
- It reduces time consumption of long process of functional and non-functional testing.
- It gives sufficient time to the developer to fix the product defects.
- It includes user point of view rather than designer or tester point of view.
- It involves examination of requirements and determination of specifications by user point of view deeply.

Techniques of Grey box Testing

- **Matrix Testing:** It defines all the used variables of a particular program. Matrix technique is a method to remove unused and uninitialized variables by identifying used variables from the program.
- **Regression Testing:** Regression testing is used to verify that modification in any part of software has not caused any adverse or unintended side effect in any other part of the software.
- **Orthogonal Array Testing or OAT:** The purpose of this testing is to cover maximum code with minimum test cases. Test cases are designed in a way that can cover maximum code as well as GUI functions with a smaller number of test cases.

Cont.

- **Pattern Testing:** Pattern testing is applicable to such type of software that is developed by following the same pattern of previous software. In these type of software possibility to occur the same type of defects. Pattern testing determines reasons of the failure so they can be fixed in the next software.
- Usually, automated software testing tools are used in Grey box methodology to conduct the test process. Stubs and module drivers provided to a tester to relieve from manually code generation.

Agile Testing

- Agile testing is an **iterative and incremental method,** and the necessities, which develop during the cooperation between the **customer** and **self-establish teams**.
- In agile testing, the word "**Agile**" primarily signifies something that can be performed quickly and immediately, but also in the area of **software development**.
- The core-functional agile team implements it in order to test the software product and its several modules. The implementation of agile testing makes sure to deliver a high quality product as bugs or defects get deleted in the initial stage of the project itself.

Principles of Agile Testing

Constant Response:

- The implementation of Agile testing delivers a response or feedback on an ongoing basis. Therefore, our product can meet the business needs.
- In other words, we can say that the Product and business requirements are understood throughout the constant response.

Less Documentation

- The execution of agile testing requires less documentation as the Agile teams or all the test engineers use a reusable specification or a checklist. And the team emphases the test rather than the secondary information.

Continuous Testing

- The agile test engineers execute the testing endlessly as this is the only technique to make sure that the constant improvement of the product.

Cont.

Customer Satisfaction

- In any project delivery, customer satisfaction is important as the customers are exposed to their product throughout the development process.
- As the development phase progresses, the customer can easily modify and update requirements. And the tests can also be changed as per the updated requirements.

Easy and clean code

- When the bugs or defects occurred by the agile team or the testing team are fixed in a similar iteration, which leads us to get the easy and clean code.

Involvement of the entire team

- As we know that, the testing team is the only team who is responsible for a testing process in the software development lifecycle. But on the other hand, in agile testing, the business analyst and the developers can also test the application or the software.

Cont.

Test-Driven

- While doing the agile testing, we need to execute the testing process during the implementation. This helps us to decrease the development time.

Quick response

- In each iteration of agile testing, the business team is involved. Therefore, we can get continuous feedback that helps us to reduces the time of feedback response on development work.

Adhoc Testing

- Adhoc testing is also known as **Monkey testing and Gorilla testing**.
- It is negative testing because we will test the application against the client's requirements.
- Aims at identifying errors that may arise due to wrong use of the software. *E.g what happens of a user inputs numbers in a field that is only supposed to have text?*
- When the end-user is using the application randomly, he/she may see a bug, but the professional test engineer uses the software systematically, so he/she may not find the same bug.

Cont.

Advantages of Adhoc Testing

- Adhoc testing cannot follow any process; that why it can be performed anytime in the software development life cycle.
- The test engineer can test the application in their new ways that helps us to find out many numbers of bugs as compared to the actual testing process.
- The developer can also execute adhoc testing while developing the module that helps them to code in a better way.
- When the in-depth testing is required in less time, adhoc testing can be performed and also deliver the quality product on time.
- Adhoc testing does not require any documentation; that's why tester can test the application with more concentration without worrying about the formal documentation.

Disadvantages of Adhoc Testing

- Adhoc testing is dependent on the test engineer's product knowledge because he/she knows the flow of the application, so he/she knows where the application can collapse, and a new test engineer may not be that much familiar with the application.
- Sometimes reproducing the bug is difficult because, in this testing, we did not follow any planning.

Assignment

- To be submitted on or before 17th May 2023
- Discuss the concept of software test automation highlighting important aspects, tools, and benefits (30 marks)

Software Systems are bound to have bugs that's why testing is important

How people react differently to a single word:

“Bug”

Tester

Developer

Manager

Developers will understand the pain...