Assignment No. 5

Aim:
Testing of project problem statement using generated test data (using mathematical models, GUI, Function testing principles, if any) selection and appropriate use of testing tools, testing of UML diagram's reliability.

Theory:
Software Testing:

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also to assess the feature of A software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process.

Verification:

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.
Basics of Software Testing:

There are two basics of software testing:**Black-box testing and White-box testing.**

Levels Of Testing:
1. Black-box Testing:
Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

2. Demonstrate:
The processes of finding major features work with typical inputs.

3. VERIFY:
It is the process of finding faults in the requirements, Design.

4. Validate:
It is the process of finding as many faults in the requirement and design.

5. Prevent:
To avoid errors in the development of requirements, design and implementation.

Principles of Testing:
1. Testing should be based on user requirements.

2. Testing time and resources are limited.

3. Exhaustive time is impossible.

4. Use effective resources to test.

5. Test planning should be done early.

6. Testing should begin in small and progress towards the testing in large.

Validation and Verification:

1. SOFTWARE VERIFICATION:
It is the process of evaluation a system or component to determine whether the product of given development phase satisfy the condition imposed at the start of the phase.

- It is a static process.
- It does not involve any code and is human based checking.
- It uses methods like inspections, walk through, desk checking etc.
- It can catch errors that validations cannot.

2. SOFTWARE VALIDATION:
It is the process of evaluation a system or component during or at the end of the development process to determine to determine whether it specifies the specified requirements. It involves executing the actual software. it is a computer based testing process.

- It is a dynamic process.
- It involves executing of code as well as human based execution of program.
- It uses methods like Black box and white box testing.
- It can catch errors that verification cannot catch.

Note that verification and validation (VV) are complementary to each other.

Basics of Software Testing:
There are two basics of software testing:

(a) Black-box testing.

(b) White-box testing.

Black-box testing:

Black-box testing treats the software as a "black box", examining functionality without any knowledge of internal implementation. The tester is only aware of what the software is supposed to do, not how it does it. Black-box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, state transition tables, decision table testing, fuzz testing, model-based testing, use case testing, exploratory testing and specification-based testing.

Specification-based testing aims to test the functionality of software according to the applicable requirements .This level of testing usually requires thorough test cases to be provided to the tester, who then can simply verify that for a given input, the output value (or behavior), either ”is” or ”is not” the same as the expected value specified in the test case. Test cases are built around specifications and requirements, i.e., what the application is supposed to do. It uses external descriptions of the software, including specifications, requirements, and designs to derive test cases. These tests can be functional or non-functional, though usually functional.

Specification-based testing may be necessary to assure correct functionality, but it is insufficient to guard against complex or high-risk situations.

One advantage of the black box technique is that no programming knowledge is required. Whatever biases the programmers may have had, the tester likely has a different set and may emphasize different areas of functionality.

On the other hand, black-box testing has been said to be ”like a walk in a dark labyrinth without a flashlight. Because they do not examine the source code, there are situations when a tester writes many test cases to check something that could have been tested by only one test case, or leaves some parts of the program untested.

This method of test can be applied to all levels of software testing: unit, integration, system and acceptance. It typically comprises most if not all testing at higher levels, but can also dominate unit testing as well.

White-box testing:

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT).

While white-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. It can test paths within a unit, paths between units during integration, and between subsystems during a system – level test. Though this method of test design can uncover many errors or problems, it might not detect unimplemented parts of the specification or missing requirements.
Techniques used in white-box testing include:
• API testing (application programming interface) - testing of the application using public and private API’s
• Code coverage - creating tests to satisfy some criteria of code coverage (e.g., the test designer can create tests to cause all statements in the program to be executed at least once)

• Fault injection methods - intentionally introducing faults to gauge the efficacy of testing strategies
• Mutation testing methods
• Static testing methods

Code coverage tools can evaluate the completeness of a test suite that was created with any method, including black-box testing. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested. Code coverage as a software metric can be reported as a percentage for:
• Function coverage, which reports on functions executed
• Statement coverage, which reports on the number of lines executed to complete the test

Types of testing:
There are following types of testing:

- Unit Testing
- Integration Testing
- Functional Testing
- System Testing
- Stress Testing
- Performance Testing
- Usability Testing
- Acceptance Testing
- Regression Testing
- Beta Testing

Unit Testing:

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

Integration Testing:

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

Functional Testing:

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box-testing.

System Testing:

System testing is the testing to ensure that by putting the software in different environments(e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box-testing.

Stress testing:

Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

Performance Testing:

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box-testing.

Usability Testing:

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

Acceptance Testing:

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

Regression Testing:

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

Beta Testing:

Beta testing is the testing which is done by end users, a team outside development, or publicly releasing full pre - version of the product which is known as beta version. The aim of beta testing is to cover unexpected errors. It falls under the class of black box testing.