

ISO 9001:2008 Certified Institute

**JAVA INSTITUTE FOR ADVANCED TECHNOLOGY**

Department of Examinations



|  |  |
| --- | --- |
| COURSE(S) – (LEADING TO) | PROFESSIONAL HIGHER DIPLOMA IN SOFTWARE ENGINEERING |
| ASSIGNMENT STARTING DATE | 05th May 2020 |
| ASSIGNMENT CLOSING DATE | 12th May 2020 |
| UNIT NAME | SOFTWARE ENGINEERING II (SOFTWARE TESTING, QUALITY ASSURANCE AND MAINTENANCE) |
| UNIT ID | HF2W 04 |
| ASSIGNMENT ID | HF2W 04/AS/04 |
| DESCRIPTION | Software test design and techniques ( Individual Assignment) |
| DURATION | 1 Day |

**GUIDE LINES FOR CANDIDATES**

Students should describe step by step testing process which has specific steps to be executed in a definite sequence to ensure that the quality goals have been met. And student will understand each activity is carried out in a planned and systematic way.

|  |  |
| --- | --- |
| NAME : | KANDAGE DON ISHAN VIHANGA VIMUKTHI  .................................................................................................................................... (BLOCK CAPITALS) |
| NIC : | 960263812v  .................................................................................................................................... |
| SCN NO : | 177646148  .................................................................................................................................... |

Java Institute for Advanced Technology Sri Lanka

1. What is the best testing model [05 marks]

Waterfall Model

1. What's the difference between alpha and beta testing?[10 marks]

Alpha Testing is conducted by a team of highly skilled testers at development site whereas Beta Testing is always conducted in Real-Time environment by customers or end-users at their own site.

Alpha testing requires lab environment or testing environment, whereas Beta testing doesn’t require any lab environment or testing environment.

Since Alpha Testing is done onsite therefore developers as well as business analysts are involved with the testing team whereas in Beta Testing developers and business analysts are not at all involved.

Beta testers can be naive or proficient end users of software product but alpha testers are always high skilled professional testers.

Alpha Testing involves both black-box testing as well as white box testing. Beta Testing is always a black box testing or functional testing.

Alpha Testing is done before the launch of software product into the market whereas Beta Testing is done at the time of software product marketing.

Alpha Testing is conducted in the presence of developers and in the absence of end-users whereas for Beta Testing this is exactly reversed.

Alpha testing is to ensure the quality of the product before moving to Beta testing. Beta testing also concentrates on quality of the product, but gathers users input on the product and ensures that the product is ready for real-time users.

Reliability and security testing are not performed in-depth Alpha Testing Reliability, Security, Robustness are checked during Beta Testing

1. What's the difference between System testing and Acceptance testing?[10 marks]

|  |  |  |
| --- | --- | --- |
| **S.no.** | **System Testing** | **Acceptance Testing** |
| 1 | System testing is performed to test end to end functionality of the software. | Acceptance testing is performed to test whether the software is conforming specified requirements and user requirements or not. |
| 2 | Only developers and testers can perform System testing. | It can be performed by testers, stakeholders and costumers. |
| 3 | It can be both non-functional and functional testing. | It can be only functional testing. |
| 4 | In System testing, we test the performance of the whole system. | In Acceptance testing, we test whether the system is conforming requirements or not. |
| 5 | System testing uses demo input values that are selected by the testing team. | Acceptance testing uses the actual real-time input values provided by the user. |
| 6 | In this testing, we include the testing of complete specification including software and hardware, memory and number of users. | Here we test whether the software is fulfilling all the needs of the user or not. |
| 7 | System Testing is a combination of System Testing and Integration testing. | Acceptance Testing is a combination of alpha testing and beta testing. |
| 8 | It is performed before the Acceptance testing. | It is performed after the System testing. |
| 9 | System testing involves load and stress testing under non-functional testing. | Acceptance testing involves boundary value analysis, equivalence portioning and decision table under functional testing. |
| 10 | The defects found in system testing are considered to be fixed. | The defects found in acceptance testing are considered as product failure. |

1. What are the advantage of unit testing?[10 marks]

Reduces Defects in the Newly developed features or reduces bugs when changing the existing functionality.

Reduces Cost of Testing as defects are captured in very early phase.

Improves design and allows better refactoring of code.

Unit Tests, when integrated with build gives the quality of the build as well.

1. What is Software Testing Life Cycle (STLC) and if phases explain them?[10 marks]

Software Testing Life Cycle refers to a testing process which has specific steps to be executed in a definite sequence to ensure that the quality goals have been met. In the STLC process, each activity is carried out in a planned and systematic way. Each phase has different goals and deliverables. Different organizations have different phases in STLC; however, the basis remains the same.

Below are the phases of STLC:

Requirements phase

Planning Phase

Analysis phase

Design Phase

Implementation Phase

Execution Phase

Conclusion Phase

Closure Phase

#1. Requirement Phase:

During this phase of STLC, analyze and study the requirements. Have brainstorming sessions with other teams and try to find out whether the requirements are testable or not. This phase helps to identify the scope of the testing. If any feature is not testable, communicate it during this phase so that the mitigation strategy can be planned.

#2. Planning Phase:

In practical scenarios, Test planning is the first step of the testing process. In this phase, we identify the activities and resources which would help to meet the testing objectives. During planning we also try to identify the metrics, the method of gathering and tracking those metrics.

On what basis the planning is done? Only requirements?

The answer is NO. Requirements do form one of the bases but there are 2 other very important factors which influence test planning. These are:

– Test strategy of the organization.

– Risk analysis / Risk Management and mitigation.

#3. Analysis Phase:

This STLC phase defines “WHAT” to be tested. We basically identify the test conditions through the requirements document, product risks, and other test bases. The test condition should be traceable back to the requirement.

There are various factors which affect the identification of test conditions:

– Levels and depth of testing

– The complexity of the product

– Product and project risks

– Software development life cycle involved.

– Test management

– Skills and knowledge of the team.

– Availability of the stakeholders.

We should try to write down the test conditions in a detailed way. For example, for an e-commerce web application, you can have a test condition as “User should be able to make a payment”. Or you can detail it out by saying “User should be able to make payment through NEFT, debit card, and credit card”.

The most important advantage of writing the detailed test condition is that it increases the test coverage since the test cases will be written on the basis of the test condition, these details will trigger to write more detailed test cases which will eventually increase the coverage.

Also, identify the exit criteria of the testing, i.e determine some conditions when you will stop the testing.

#4. Design Phase:

This phase defines “HOW” to test. This phase involves the following tasks:

– Detail the test condition. Break down the test conditions into multiple sub-conditions to increase coverage.

– Identify and get the test data

– Identify and set up the test environment.

– Create the requirement traceability metrics

– Create test coverage metrics.

#5. Implementation Phase:

The major task in this STLC phase is of creation of the detailed test cases. Prioritize the test cases also identify which test case will become part of the regression suite. Before finalizing the test case, It is important to carry out the review to ensure the correctness of the test cases. Also, don’t forget to take the sign off of the test cases before actual execution starts.

If your project involves automation, identify the candidate test cases for automation and proceed for scripting the test cases. Don’t forget to review them!

#6. Execution Phase:

As the name suggests, this is the Software Testing Life Cycle phase where the actual execution takes place. But before you start your execution, make sure that your entry criterion is met. Execute the test cases, log defects in case of any discrepancy. Simultaneously fill your traceability metrics to track your progress.

#7. Conclusion Phase:

This STLC phase concentrates on the exit criteria and reporting. Depending on your project and stakeholders choice, you can decide on reporting whether you want to send out a daily report of the weekly report, etc.

There are different types of reports ( DSR – Daily status report, WSR – Weekly status reports) which you can send, but the important point is, the content of the report changes and depends upon whom you are sending your reports.

If Project managers belong to testing background then they are more interested in the technical aspect of the project, so include the technical things in your report ( number of test cases passed, failed, defects raised, severity 1 defects, etc.).

But if you are reporting to upper stakeholders, they might not be interested in the technical things so report them about the risks that have been mitigated through the testing.

#8. Closure Phase:

Tasks for the closure activities include the following:

– Check for the completion of the test. Whether all the test cases are executed or mitigated deliberately. Check there is no severity 1 defects opened.

– Do lessons learned meeting and create lessons learned document. ( Include what went well, where are the scope of improvements and what can be improved)

1. What are the type of test model are available?[05 marks]

Waterfall Model

V Model

Agile Model

Spiral Model

Iterative Model

1. What is the testing level and write down the various test levels available[15 marks]

Levels of testing include different methodologies that can be used while conducting software testing. The main levels of software testing are −

* Functional Testing
* Non-functional Testing

Functional Testing

This is a type of black-box testing that is based on the specifications of the software that is to be tested. The application is tested by providing input and then the results are examined that need to conform to the functionality it was intended for. Functional testing of a software is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.

There are five steps that are involved while testing an application for functionality.

|  |  |
| --- | --- |
| **Steps** | **Description** |
| I | The determination of the functionality that the intended application is meant to perform. |
| II | The creation of test data based on the specifications of the application. |
| III | The output based on the test data and the specifications of the application. |
| IV | The writing of test scenarios and the execution of test cases. |
| V | The comparison of actual and expected results based on the executed test cases. |

An effective testing practice will see the above steps applied to the testing policies of every organization and hence it will make sure that the organization maintains the strictest of standards when it comes to software quality.

Unit Testing

This type of testing is performed by developers before the setup is handed over to the testing team to formally execute the test cases. Unit testing is performed by the respective developers on the individual units of source code assigned areas. The developers use test data that is different from the test data of the quality assurance team.

The goal of unit testing is to isolate each part of the program and show that individual parts are correct in terms of requirements and functionality.

Limitations of Unit Testing

Testing cannot catch each and every bug in an application. It is impossible to evaluate every execution path in every software application. The same is the case with unit testing.

There is a limit to the number of scenarios and test data that a developer can use to verify a source code. After having exhausted all the options, there is no choice but to stop unit testing and merge the code segment with other units.

Integration Testing

Integration testing is defined as the testing of combined parts of an application to determine if they function correctly. Integration testing can be done in two ways: Bottom-up integration testing and Top-down integration testing.

|  |  |
| --- | --- |
| **Sr.No.** | **Integration Testing Method** |
| 1 | **Bottom-up integration**  This testing begins with unit testing, followed by tests of progressively higher-level combinations of units called modules or builds. |
| 2 | **Top-down integration**  In this testing, the highest-level modules are tested first and progressively, lower-level modules are tested thereafter. |

In a comprehensive software development environment, bottom-up testing is usually done first, followed by top-down testing. The process concludes with multiple tests of the complete application, preferably in scenarios designed to mimic actual situations.

System Testing

System testing tests the system as a whole. Once all the components are integrated, the application as a whole is tested rigorously to see that it meets the specified Quality Standards. This type of testing is performed by a specialized testing team.

System testing is important because of the following reasons −

* System testing is the first step in the Software Development Life Cycle, where the application is tested as a whole.
* The application is tested thoroughly to verify that it meets the functional and technical specifications.
* The application is tested in an environment that is very close to the production environment where the application will be deployed.
* System testing enables us to test, verify, and validate both the business requirements as well as the application architecture.

Regression Testing

Whenever a change in a software application is made, it is quite possible that other areas within the application have been affected by this change. Regression testing is performed to verify that a fixed bug hasn't resulted in another functionality or business rule violation. The intent of regression testing is to ensure that a change, such as a bug fix should not result in another fault being uncovered in the application.

Regression testing is important because of the following reasons −

* Minimize the gaps in testing when an application with changes made has to be tested.
* Testing the new changes to verify that the changes made did not affect any other area of the application.
* Mitigates risks when regression testing is performed on the application.
* Test coverage is increased without compromising timelines.
* Increase speed to market the product.

Acceptance Testing

This is arguably the most important type of testing, as it is conducted by the Quality Assurance Team who will gauge whether the application meets the intended specifications and satisfies the client’s requirement. The QA team will have a set of pre-written scenarios and test cases that will be used to test the application.

More ideas will be shared about the application and more tests can be performed on it to gauge its accuracy and the reasons why the project was initiated. Acceptance tests are not only intended to point out simple spelling mistakes, cosmetic errors, or interface gaps, but also to point out any bugs in the application that will result in system crashes or major errors in the application.

By performing acceptance tests on an application, the testing team will reduce how the application will perform in production. There are also legal and contractual requirements for acceptance of the system.

Alpha Testing

This test is the first stage of testing and will be performed amongst the teams (developer and QA teams). Unit testing, integration testing and system testing when combined together is known as alpha testing. During this phase, the following aspects will be tested in the application −

* Spelling Mistakes
* Broken Links
* Cloudy Directions
* The Application will be tested on machines with the lowest specification to test loading times and any latency problems.

Beta Testing

This test is performed after alpha testing has been successfully performed. In beta testing, a sample of the intended audience tests the application. Beta testing is also known as **pre-release testing**. Beta test versions of software are ideally distributed to a wide audience on the Web, partly to give the program a "real-world" test and partly to provide a preview of the next release. In this phase, the audience will be testing the following −

* Users will install, run the application and send their feedback to the project team.
* Typographical errors, confusing application flow, and even crashes.
* Getting the feedback, the project team can fix the problems before releasing the software to the actual users.
* The more issues you fix that solve real user problems, the higher the quality of your application will be.
* Having a higher-quality application when you release it to the general public will increase customer satisfaction.

Non-Functional Testing

This section is based upon testing an application from its non-functional attributes. Non-functional testing involves testing a software from the requirements which are nonfunctional in nature but important such as performance, security, user interface, etc.

Some of the important and commonly used non-functional testing types are discussed below.

Performance Testing

It is mostly used to identify any bottlenecks or performance issues rather than finding bugs in a software. There are different causes that contribute in lowering the performance of a software −

* Network delay
* Client-side processing
* Database transaction processing
* Load balancing between servers
* Data rendering

Performance testing is considered as one of the important and mandatory testing type in terms of the following aspects −

* Speed (i.e. Response Time, data rendering and accessing)
* Capacity
* Stability
* Scalability

Performance testing can be either qualitative or quantitative and can be divided into different sub-types such as **Load testing** and **Stress testing**.

Load Testing

It is a process of testing the behavior of a software by applying maximum load in terms of software accessing and manipulating large input data. It can be done at both normal and peak load conditions. This type of testing identifies the maximum capacity of software and its behavior at peak time.

Most of the time, load testing is performed with the help of automated tools such as Load Runner, AppLoader, IBM Rational Performance Tester, Apache JMeter, Silk Performer, Visual Studio Load Test, etc.

Virtual users (VUsers) are defined in the automated testing tool and the script is executed to verify the load testing for the software. The number of users can be increased or decreased concurrently or incrementally based upon the requirements.

Stress Testing

Stress testing includes testing the behavior of a software under abnormal conditions. For example, it may include taking away some resources or applying a load beyond the actual load limit.

The aim of stress testing is to test the software by applying the load to the system and taking over the resources used by the software to identify the breaking point. This testing can be performed by testing different scenarios such as −

* Shutdown or restart of network ports randomly
* Turning the database on or off
* Running different processes that consume resources such as CPU, memory, server, etc.

Usability Testing

Usability testing is a black-box technique and is used to identify any error(s) and improvements in the software by observing the users through their usage and operation.

According to Nielsen, usability can be defined in terms of five factors, i.e. efficiency of use, learn-ability, memory-ability, errors/safety, and satisfaction. According to him, the usability of a product will be good and the system is usable if it possesses the above factors.

Nigel Bevan and Macleod considered that usability is the quality requirement that can be measured as the outcome of interactions with a computer system. This requirement can be fulfilled and the end-user will be satisfied if the intended goals are achieved effectively with the use of proper resources.

Molich in 2000 stated that a user-friendly system should fulfill the following five goals, i.e., easy to Learn, easy to remember, efficient to use, satisfactory to use, and easy to understand.

In addition to the different definitions of usability, there are some standards and quality models and methods that define usability in the form of attributes and sub-attributes such as ISO-9126, ISO-9241-11, ISO-13407, and IEEE std.610.12, etc.

UI vs Usability Testing

UI testing involves testing the Graphical User Interface of the Software. UI testing ensures that the GUI functions according to the requirements and tested in terms of color, alignment, size, and other properties.

On the other hand, usability testing ensures a good and user-friendly GUI that can be easily handled. UI testing can be considered as a sub-part of usability testing.

Security Testing

Security testing involves testing a software in order to identify any flaws and gaps from security and vulnerability point of view. Listed below are the main aspects that security testing should ensure −

* Confidentiality
* Integrity
* Authentication
* Availability
* Authorization
* Non-repudiation
* Software is secure against known and unknown vulnerabilities
* Software data is secure
* Software is according to all security regulations
* Input checking and validation
* SQL insertion attacks
* Injection flaws
* Session management issues
* Cross-site scripting attacks
* Buffer overflows vulnerabilities
* Directory traversal attacks

Portability Testing

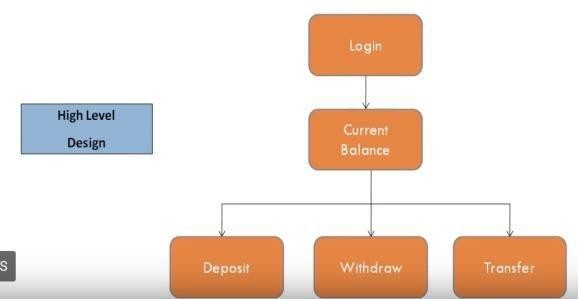
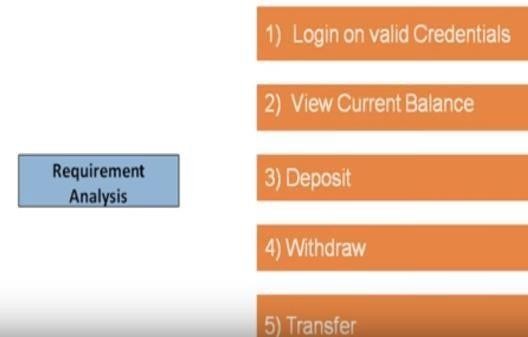
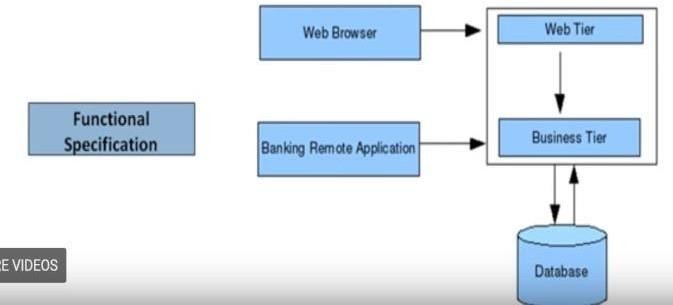
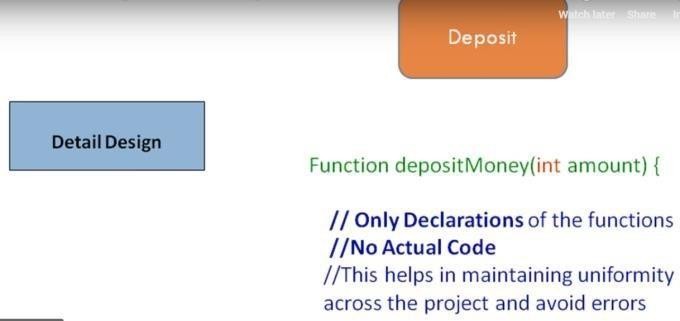
Portability testing includes testing a software with the aim to ensure its reusability and that it can be moved from another software as well. Following are the strategies that can be used for portability testing −

* Transferring an installed software from one computer to another.
* Building executable (.exe) to run the software on different platforms.

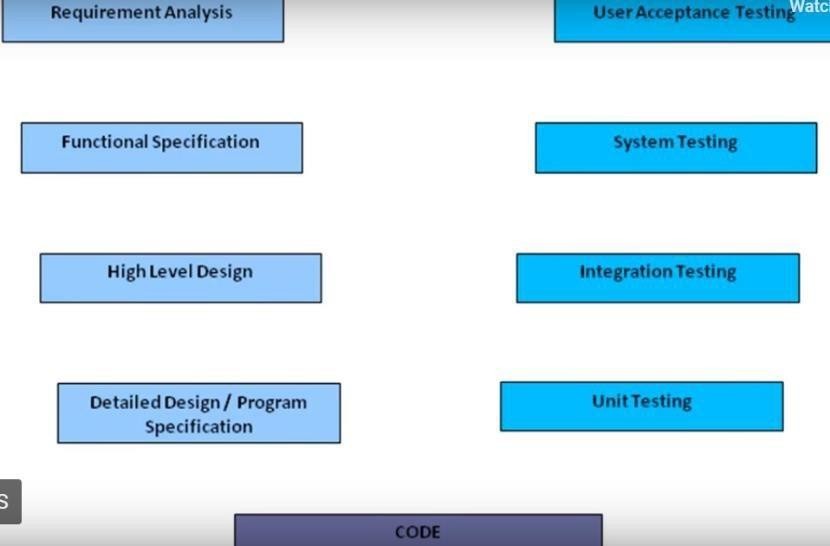
Portability testing can be considered as one of the sub-parts of system testing, as this testing type includes overall testing of a software with respect to its usage over different environments. Computer hardware, operating systems, and browsers are the major focus of portability testing. Some of the pre-conditions for portability testing are as follows −

* Software should be designed and coded, keeping in mind the portability requirements.
* Unit testing has been performed on the associated components.
* Integration testing has been performed.
* Test environment has been established.

1. You are haring to develop a banking application. Bellow mention the Functionality of a system. (SRS- requirement and functions )



Pseudo code is design for each module



V model

Explain how each tasting phases take place according to your banking application. (25 marks)

**Types of Requirements**

1. **Business requirements**: They are high-level requirements that are taken from the business case from the projects.

For example, a mobile banking service system provides banking services to Southeast Asia. The business requirement that is decided for India is account summary and fund transfer while for China account summary and bill payment is decided as a business requirement

|  |  |
| --- | --- |
| Country | Company providing Banking Functionalities or services |
| India | Account Summary and Fund Transfer |
| China | Account Summary and Bill Payment |

1. **Architectural and Design requirements**: These requirements are more detailed than business requirements. It determines the overall design required to implement the business requirement.

For our educational organization the architectural and design use cases would be login, course detail, etc. The requirement would be as shown below.

|  |  |
| --- | --- |
| Banking use case | Requirement |
| Bill Payment | This use case describes how a customer can login into net banking and use the Bill Payment Facility.  The customer will can see a dashboard of outstanding bills of registered billers. He can add, modify, and delete a biller detail. The customer can configure SMS, email alerts for different billing actions. He can see history of past paid bills.  The actors starting this use case are bank customers or support personnel. |

1. **System and Integration requirements**: At the lowest level, we have system and integration requirements. It is detailed description of each and every requirement. It can be in form of user stories which is really describing everyday business language. The requirements are in abundant details so that developers can begin coding.

Here in example of Bill Payment module where requirement will be mentioned for adding a Biller

|  |  |
| --- | --- |
| Bill Payment | Requirements |
| Add Billers | * Utility Provider Name * Relationship/Customer Number * Auto Payments – Yes/No * Pay Entire Bill – Yes/No * Auto Payment Limit – Do not pay if Bill is over specified amount |

Sometimes for some project you might not receive any requirements or documents to work with. But still there are other sources of requirements that you can consider for the requirement or information, so that you can base your software or test design on these requirements.

1. What is State Transition Testing? And when it use? [5marks]

State Transition testing is defined as the software testing technique in which changes in input conditions cause's state changes in the Application under Test (AUT).

It is a black box testing technique in which the tester analyzes the behavior of an application under test for different input conditions in a sequence. In this technique, tester provides both positive and negative input test values and record the system behavior.

It is the model on which the system and the tests are based. Any system where you get a different output for the same input, depending on what has happened before, is a finite state system.

This can be used when a tester is testing the application for a finite set of input values.

When the tester is trying to test sequence of events that occur in the application under test. I.e., this will allow the tester to test the application behavior for a sequence of input values.

When the system under test has a dependency on the events/values in the past.

1. What is the test report and explain the it included things [5 marks]

Test Report is a document which contains

A summary of test activities and final test results

An assessment of how well the Testing is performed

Based on the test report, the stakeholders can

Evaluate the quality of the tested product

Make a decision on the software release. For example, if the test report informs that there’re many defects remaining in the product, the stakeholder can delay the release until all the defects are fixed.