**Software Testing Assignment**

**Module-1**

1. **What is SDLC**

SDLC means Software Development Life Cycle.

It is a structured process which is used to design, develop, and test good quality software.

The goal of it is to deliver high quality, maintainable software that meets the user’s requirements.

There are six phases of it.

(1)Requirements Collection/Gathering: Establish Customer Needs

(2)Analysis : Model & Specify the requirements- “What”

(3)Design: Model & Specify a Solution – “Why”

(4)Implementation: Construct a Solution In Software

(5)Testing: Validate the solution against the requirements

(6)Maintenance: Repair defects & adapt the solution to the new

requirements

1. **What is Software Testing**

Software testing is process used to identify the correctness, completeness, & quality of developed computer software.

Testing Activities

Planning & control.

Chossing that conditions.

Designing test cases.

Checking results.

Evaluating completion criteria.

Reporting on the testing process and system under test.

Finalizing or closure

Testing also includes reviewing of document.

7 keys Principal of testing

1. Testing shows presence of defects.
2. Exhaustive testing is possible.
3. Early testing
4. Defect clustering
5. The pesticide paradox
6. Testing is context dependent
7. Absence of errors fallacy
8. **What is Agile methodology ?**

Agile means move quickly.

It is a proper way of managing the project with breaking them into smaller phases

which is iteration.

It basically focus on flexibility of the project which we can change and improve the

team work as per requirements.

This allows team to adapt to changes quickly & provide customer value faster.

Major companies like Facebook, Google, Amazon use agile because of its adaptability & customer focused approach.

It focuses on working together & making constant improvements.

Team plans ,work on project ,&then review this how cycle works.

1. **What is SRS ?**

SRS means software requirements specification

It is a complete description of the behaviour of the system to be developed.

It includes a set of use cases that describe all of the interactions that the users will have with the software.

There are types of requirements which are needed

1. Customer Requirements
2. Functional Requirements
3. Non-Functional Requirements
4. Customer Requirements: **Performance, Functionality, Usability, Reliability, Maintainability, Cost, Security, Scalability.**

**Customer satisfaction & loyalty is more importants.so we have to work upon customer requirements.**

1. Functional Requirements

Functional Requirements are very important system requirements in the system design process.

Ex: like an email we can able to receive & send emails, create folders ,filter & pin emails, can attach different kind of attachments.

1. Non-functional Requirements

It includes quality attributes of a software product, focusing on how the system performs rather than what it does.

Non functional requirements like performance, usability, security, scalability

maintainability, reliability.

It can be divided into following categories

1. Usability 2. Reliability 3. Performance 4. Security

1. **What is OOP?**

OOP stands for object oriented programming.

Identifying objects and assigning responsibilities to these objects.

Objects communicate to other objects by sending messages

Messages are received by the methods of an object

Object-oriented programming has a web of interacting objects, each house-keeping its own state.

Object oriented programming like C++ & JAVA.

1. **Write basic concept of OOPS.**

1. Object

2. Class

3. Encapsulation

4. Inheritance

5. Polymorphism

(i) Overriding

(ii) Overloading

6. Abstraction

1. **What is Object** .

Object is instants of class.

Object have state & behaviour.

Ex:- car, table, laptop etc..

Each object have different different types of colour,shape,size,design.

An object is anything to which a concept applies.

That is both data and function that operate on data are bundled as a unit called as object.

1. **What is class ?**

Class is a structure in which we can have member functions & member variables are there.

When you define a class, you define a blueprint for an object.

A class represents an abstraction of the object and abstracts the properties and behaviour of that object.

Class can be considered as the blueprint or definition or a template for an object and

describe the properties and behaviour of that object, but without any actual existence

An object is a particular instance of a class which has actual existence and there can be many objects (or instances) for a class.

In the case of a car or laptop, there will be a blueprint or design created first and then the actual car or laptop will be built based on that. We do not actually buy these blueprints but the actual objects.

1. **What is Encapsulation ?**

To wrapping data into single unit is called encapsulation.

Encapsulation is the practice of including in an object everything it needs hidden from other objects. The internal state is usually not accessible by other objects.

Encapsulation is placing the data and the functions that work on that data in the same place.

While working with procedural languages, it is not always clear which functions work on which variables but object-oriented programming provides you framework to place the data and the relevant functions together in the same object.

In Java, everything is enclosed within a class or interface, unlike languages such as C and C++, where we can have global variables outside classes.

We can expose our operations hiding the details of what is needed to perform that operation.

1. **What is inheritance ?**

To access property one class to another class is called inheritance.

This is a very important concept of object-oriented programming since this feature helps to reduce the code size.

Inheritance describes the relationship between two classes.

A class can get some of its characteristics from a parent class and then add unique features of its own.

There are 5 types of it

1.Single

2.Multilevel

3.Multiple

4.Hierarchical

5.Hybrid

1. **What is polymorphism ?**

Same function name but having different functionalities is called polymorphism.

It means having many forms.

**It allows different objects to respond to the same message in different ways, the**

**response specific to the type of the object.**

Poly refers too many.

**The ability to change form is known as polymorphism.**

There is two types of polymorphism in Java.

**(1)Overloading**

**(2)Overriding**

**(1)Overloading**

The concept of overloading is also a branch of polymorphism.

When the exiting operator or function is made to operate on new data type, it is said

to be overloaded.

The same method name (method overloading) or operator symbol (operator

Overloading) can be used in different contents.

In method overloading, multiple methods having same name can appear in a class

But with a with different signature.

**(2)Overriding**

Overriding is defining a method in a subclass with the same name and type signature

as a method in its super class and when this subclass instance appears in the super

class context like ..

parent p=New Child()

and when we execute an overridden method as p.myMethod(), the subtype’s version

of that method is executed.

Here, the actual method called will depend on the object at runtime, not the

reference type.

type. Consider an example class Shape with a draw() method.

It can have subclasses Circle and Square

An object of Circle or Square can be assigned to a Shape reference as

Shape s=new Circle();

While executing draw() on the Shape reference, it will draw a Circle or Square based

on the actual object assigned to it at runtime.

1. **Draw Usecase on Online book shopping.**

<https://drive.google.com/file/d/1auS9W_FhRbARX-YB0RgeRZfMzzApbcZm/view?usp=sharing>

1. **Draw Usecase on online bill payment system(paytm).**

https://drive.google.com/file/d/1tjTDUkV8-HZ2jn43HJo6M5\_sWwENMib-/view?usp=sharing

**14. Write SDLC phases with basic introduction.**

There are six phases of it.

(1)Requirements Collection/Gathering: Establish Customer Needs

(2)Analysis : Model & Specify the requirements- “What”

(3)Design: Model & Specify a Solution – “Why”

(4)Implementation: Construct a Solution In Software

(5)Testing: Validate the solution against the requirements

(6)Maintenance: Repair defects & adapt the solution to the new requirements

1. **Requirements Collection**

Features, Usage scenarios,

Requirements may be in documented in Written form it maybe incomplete or incorrect although we can change it as well.

Early prototyping like UI , it can help us to clarify the requirements.

There are Functional and Non-Functional Requirements as well.

1. **Analysis**

The analysis phase defines the requirements of the system.

It defines the problem that the customer is trying to solve.

We have to figure it out that which language is suitable for this app.

Who are the best developers, tester, designer it means which is the perfect team to do this project we have to figure it out as well.

1. **Design**

Design Architecture Document, Implementation Plan, Critical Priority Analysis, Test plan. The design team have to decide which design will be look good or they will have to work upon client’s requirement.

1. **Implementation**

In this the team can implement on work what they have done

They can change upon client Requirement.

Like client wants narrowly design now first he wanted other design so we have to change it .

1. **Testing**

Quality is more important, many companies have not learned that quality is important & deliver more claimed functionality but at a lower quality level.

It is much easier to explain to a customer why there is a missing feature than to explain to a customer why the product lacks quality.

Customer satisfaction is more important .so we have to solve the bugs from the main feature what he wants. We can’t say that the software is fully bug free, we can work on it until the next update.

1. **Maintenance**

Software maintenance is also one of the phases of SDLC.

This phase which comes after deployment of the software into the field.

The developing organization or team will have some mechanism to document and track defects.

Updating all analysis & design and user documentation.

**15. Explain phases of waterfall model.**

Requirements must be “frozen” to early in the life cycle.

Requirements are validated too late.  
Requirements are very well documented, clear and fixed.

Product definition is stable.

Technology is understood and is not dynamic

When the project is short it is useful.

**PROS**

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

Easy to arrange tasks.

Process and results are well documented.

**Cons**

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 on going projects.

Cannot accommodate changing requirements

No working software is produced until late in the life cycle.

Adjusting scope during the life cycle can end a project.

**16.Write phases of spiral model.**

Spiral Model is very widely used in the software industry as it is in synch with the natural development process of any product i.e. learning with maturity and also involves minimum risk.It is use for medium to high-risk projects .

Long-term project commitment because of potential changes to economic priorities as the requirements change with time.

Requirements are complex and need evaluation to get clarity.

**Pros**

Changing requirements can be accommodated.

Requirements can be captured more accurately.

Users see the system early, Allows for extensive use of prototypes.

Development can be divided into smaller parts and more risky parts can be developed earlier which helps better risk management.

**Cons**

Management is more complex.

Process is complex, Spiral may go indefinitely.

Large number of intermediate stages requires excessive documentation

Not suitable for small or low risk projects and could be expensive for small projects.

**17. Write agile manifesto principles**

**Individuals and interactions** - in agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.

**Working software** - Demo working software is considered the best means of communication with the customer to understand their requirement, instead of just depending on documentation.

**Customer collaboration** - As the requirements cannot be gathered completely in the

beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.

**Responding to change** - agile development is focused on quick responses to change and continuous development.

**18. Explain working methodology of agile model & also write pros & cons.**

Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In agile the tasks are divided to time boxes to deliver specific features for a release.

Agile thought process had started early in the software development and started becoming popular with time due to its flexibility and adaptability.

Agile SDLC model is a combination of iterative and incremental process models with

focus on process adaptability and customer satisfaction by rapid delivery of working

software product.

It break the product into small incremental builds.

These builds are provided in iterations.

Each iteration typically lasts from about one to three weeks.

Every iteration involves cross functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing, and acceptance testing.

At the end of the iteration a working product is displayed to the customer and important stakeholders.

**Pros**

Is a very realistic approach to software development.

Functionality can be developed rapidly and demonstrated

Resource requirements are minimum.

Promotes teamwork and cross training.

Suitable for fixed or changing requirements & Delivers early partial working solutions.

Good model for environments that change steadily.

No planning required easy to manage gives flexibility to developers.

**Cons**

Not suitable for handling complex dependencies.

More risk of sustainability, maintainability and extensibility.

Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.

There is very high individual dependency, since there is minimum documentation generated.

Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.

**19. Draw usecase on online shopping product using COD.**

https://drive.google.com/file/d/1WYIe-Z1pAtdXGJSIQs5iqfg2Lrj3m79i/view?usp=sharing

**20.Draw usecase on Online shopping product using payment gateway.**

https://drive.google.com/file/d/1DpzcSYrvzSQPOO-rz1QsR8xlFFQ8g9w4/view?usp=sharing