What is SDLC?

Requirement Gathering and Analysis.

Planning.

Design.

Implementation.

Testing.

Deployment.

Maintenance.

What is software testing?

Testing is the process of evaluating a system or its component(s) with the intent to find that whether it satisfies the specified requirements or not.

In simple words testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements.

Software Testing is a process used to identify the correctness, completeness, and quality of developed computer software.

What is agile methodology?

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.

Agile Methods break the product into small incremental builds.

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.

What is SRS?

**SRS** stands for **Software Requirements Specification**. It is a formal document that describes what the software system should do. It acts as a **blueprint** for both the **development team** and the **testing team**.

 **Functional Requirements**

* Detailed features and behaviors the system should support
* Example: "The system shall allow users to log in using email and password."

 **Non-Functional Requirements**

* Performance, security, usability, scalability, etc.
* Example: "The system should load the dashboard within 3 seconds."

What is oops?

Identifying objects and assigning responsibilities to these objects.

Objects communicate to other objects by sending messages.

**OOP**, or **Object-Oriented Programming**, is a programming paradigm based on the concept of **"objects"**, which contain **data** and **code**

Write Basic Concepts of oops.

Class

Object

**Encapsulation**

**Abstraction**

Inheritance

Polymorphism

What is object?

Tangible Things

Roles

Incidents

Interactions

Specifications

What is class?

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 behavior of that object.

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

This doesn't actually define any data, but it does define what the class name means, that is, what an object of the class will consist of and what operations can be performed on such an object.

What is 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.

Encapsulation in Java is the process of wrapping up of data and behavior of an object into a single unit; and the unit here is a Class .

Encapsulation enables data hiding, hiding irrelevant information from the users of a class and exposing only the relevant details required by the user

What is inheritance?

Inheritance means that one class inherits the characteristics of another class. This is also called a “is a” relationship.

One of the most useful aspects of object-oriented programming is code reusability. As the name suggests Inheritance is the process of forming a new class from an existing class that is from the existing class called as base class, new class is formed called as derived class.

In general, Java supports single-parent, multiple-children inheritance and multilevel inheritance (Grandparent-> Parent -> Child) for classes and interfaces. Java supports multiple inheritances (multiple parents, single child) only through interfaces.

What is polymorphism?

Polymorphism 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.

The most important aspect of an object is its behaviour (the things it can do). A behaviour is initiated by sending a message to the object (usually by calling a method)..

The ability to change form is known as polymorphism

Write SDLC phases with basic introduction?

SDLC is a structure imposed on the development of a software product that defines the process for planning, implementation, testing, documentation, deployment, and ongoing maintenance and support. There are a number of different development models.

A Software Development Life Cycle is essentially a series of steps, or phases, that provide a model for the development and lifecycle management of an application or piece of software.

Requirements Collection/Gathering Establish Customer Needs

Analysis Model And Specify the requirements “What”

Design Model And Specify a Solution – “Why”

Implementation Construct a Solution In Software

Testing Validate the solution against the requirements

Maintenanc Repair defects and adapt the solution to the new requirements

Explain Phases of the waterfall model.

The waterfall is unrealistic for many reasons, especially.

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.

There are no ambiguous requirements.

Ample resources with required expertise are available to support the product.

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 for the customer as well as the development firms. Following are the typical uses of Spiral model:

When costs there are a budget constraint and risk evaluation is important.

For medium to high-risk projects.

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

Customer is not sure of their requirements which are usually the case.

Requirements are complex and need evaluation to get clarity.

New product line which should be released in phases to get enough customer feedback.

Significant changes are expected in the product during the development cycle.

Explain working methodology of agile model and also write pros and 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 (small time frames) 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 f lexibility and adaptability.

Pros

Is a very realistic approach to software development.

Promotes teamwork and cross training.

Functionality can be developed rapidly and demonstrated.

Resource requirements are minimum.

Suitable for fixed or changing requirements

Delivers early partial working solutions.

Good model for environments that change steadily.

Minimal rules, documentation easily employed.

Enables concurrent development and delivery within an overall.

Little or no planning required

Easy to manage

Gives flexibility to developers

Cons

Not suitable for handling complex dependencies.

More risk of sustainability, maintainability and extensibility.

An overall plan, an agile leader and agile PM practice is a must without which it will not work.

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

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

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

Transfer of technology to new team members may be quite challenging due to lack of documentation.



