<1> What is SDLC?

:-> SDLC full form is software Development Life Cycle.

The software development lifecycle (SDLC) is the cost-effective and time-efficient process that development teams use to design and build high-quality software. The goal of SDLC is to minimize project risks through forward planning so that software meets customer expectations during production and beyond.

**The 7 Phases Of SDLC (Software Development Life Cycle)**

* Planning & Requirement Gathering...
* Analysis. ...
* Design. ...
* Build...
* Testing. ...
* Deployment. ...
* Maintenance.

<2> What is Software Testing?

:-> Software Testing is the process of evaluating and verifying the functionality of software products or apps to ensure they meet expected requirements and are free from defects.

Testing is the process of evaluating a system or its component(s) with the intent to find 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 requirements.

**\*Software testing can be divided in to two steps**

## :-> Verification testing

Verification testing includes different activities such as business requirements, system requirements, design review, and code walkthrough while developing a product.

It is also known as static testing, where we are ensuring that "**we are developing the right product or not**". And it also checks that the developed application fulfilling all the requirements given by the client.

## :-> Validation testing

Validation testing is testing where tester performed functional and non-functional testing. Here **functional testing** includes [UnitTesting](https://www.javatpoint.com/unit-testing) (UT), [Integration Testing](https://www.javatpoint.com/integration-testing) (IT) and System Testing (ST), and **non-functional** testing includes User acceptance testing (UAT).

<3> What is Agile Methodology?

:-> Agile testing is software testing that follows the best practices of the Agile development framework. Agile development takes an incremental approach to development. Similarly, Agile testing includes an incremental approach to testing. In this type of software testing, features are tested as they are developed.

The **Agile software development** methodology is one of the simplest and effective processes to turn a vision for a business need into software solutions. Agile is a term used to describe software development approaches that employ continual planning, learning, improvement, team collaboration, evolutionary development, and early delivery. It encourages flexible responses to change.

The agile software development emphasizes on four core values.

1. Individual and team interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

<4> What is SRS?

:-> A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill the needs of all stakeholders (business, users).

A software requirements specification (SRS) is a comprehensive description of the intended purpose and environment for [software](https://www.techtarget.com/searchapparchitecture/definition/software) under development. The SRS fully describes what the software will do and how it will be expected to perform.

An SRS minimizes the time and effort required by developers to achieve desired goals and also minimizes the development cost. A good SRS defines how an application will interact with system hardware, other programs and human users in a wide variety of real-world situations. Parameters such as operating speed, response time, availability, [portability](https://www.techtarget.com/searchstorage/definition/portability), maintainability, [footprint](https://www.techtarget.com/whatis/definition/footprint), security and speed of recovery from adverse events are evaluated.

<5> What is OOPs?

:-> Object-Oriented Programming or OOPs refers to languages that use objects in programming. Object-oriented programming aims to implement real-world entities like inheritance, hiding, polymorphism, etc in programming. The main aim of OOP is to bind together the data and the functions that operate on them so that no other part of the code can access this data except that function.

<6> Write basic concepts of OOPs?

:-> **OOPs Concepts:**

* Class
* Objects
* Data Abstraction
* Encapsulation
* Inheritance
* Polymorphism
* Dynamic Binding
* Message Passing

<7> What is object?

:-> It is a basic unit of Object-Oriented Programming and represents the real-life entities. An Object is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated. An object has an identity, state, and behavior. Each object contains data and code to manipulate the data. Objects can interact without having to know details of each other’s data or code, it is sufficient to know the type of message accepted and type of response returned by the objects.

For example “Dog” is a real-life Object, which has some characteristics like color, Breed, Bark, Sleep, and Eats.

<8> What is class?

:-> A class is a user-defined data type. It consists of data members and member functions, which can be accessed and used by creating an instance of that class. It represents the set of properties or methods that are common to all objects of one type. A class is like a blueprint for an object.

***For Example:***Consider the Class of Cars. There may be many cars with different names and brands but all of them will share some common properties like all of them will have 4 wheels, Speed Limit, Mileage range, etc. So here, Car is the class, and wheels, speed limits, mileage are their properties.

<9> What is encapsulation?

:-> Encapsulation is defined as the wrapping up of data under a single unit. It is the mechanism that binds together code and the data it manipulates. In Encapsulation, the variables or data of a class are hidden from any other class and can be accessed only through any member function of their class in which they are declared. As in encapsulation, the data in a class is hidden from other classes, so it is also known as **data-hiding**.

<10> What is inheritance?

:-> Inheritance is an important pillar of OOP(Object-Oriented Programming). The capability of a class to derive properties and characteristics from another class is called Inheritance. When we write a class, we inherit properties from other classes. So when we create a class, we do not need to write all the properties and functions again and again, as these can be inherited from another class that possesses it. Inheritance allows the user to reuse the code whenever possible and reduce its redundancy.

<11> What is polymorphism?

:-> The word polymorphism means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form. For example, A person at the same time can have different characteristics. Like a man at the same time is a father, a husband, an employee. So the same person posses different behavior in different situations. This is called polymorphism.

<12> Draw Usecase on Online Book Shopping.

:->

<13> Draw Usecase on Online Bill payment system(paytm).

:->

<14> Write SDLC phases with basic introduction.

:-> **The 7 Phases Of SDLC (Software Development Life Cycle)**

* Planning & Requirement Gathering...
* Analysis....
* Design....
* Build...
* Testing....
* Deployment....
* Maintenance….
* Planning & Requirement Gathering...

Requirements gathering is the process of understanding what you are trying to build and why you are building it. Requirements gathering is often regarded as a part of developing software applications or of cyber-physical systems like aircraft, spacecraft, and automobiles.

* Analysis....

System analysis refers to the process of gathering data, interpreting information, identifying issues and using the results to recommend or develop possible system improvements.

* Design....

In this phase, the software design is created, which includes the overall architecture of the software, data structures, and interfaces. It has two steps:

**:- High-level design (HLD):** It gives the architecture of software products.

**:- Low-level design (LLD):** It describes how each and every feature in the product should work and every component.

* Build...

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

* Testing....

The software is thoroughly tested to ensure that it meets the requirements and works correctly.

* Deployment....

After successful testing, The software is deployed to a production environment and made available to end-users.

* Maintenance….

This phase includes ongoing support, bug fixes, and updates to the software.

There are **different methodologies** that organizations can use to implement the SDLC, such as**Waterfall, Agile, Scrum, V-Model**and**DevOps.**

<15> Explain phases of the waterfall model.

:-> There are 6 phases in a simple waterfall model: requirements, design, implementation, verification, deployment and maintenance. The waterfall method is easy to understand and manage, results in fewer production issues and is easier to keep on budget.

**The Waterfall Model has six phases:**

1. **Requirements Gathering and Analysis:** The first phase involves gathering requirements from stakeholders and analyzing them to understand the scope and objectives of the project.
2. **Design Phase:** Once the requirements are understood, the design phase begins. This involves creating a detailed design document that outlines the software architecture, user interface, and system components.
3. **Implementation and Unit Testing:** The implementation phase involves coding the software based on the design specifications. This phase also includes unit testing to ensure that each component of the software is working as expected.
4. **Integration and System Testing:** In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects.
5. **Deployment:** Once the software has been tested and approved, it is deployed to the production environment.

**6. Maintenance:** The final phase of the Waterfall Model is maintenance, which involves fixing any issues that arise after the software has been deployed and ensuring that it continues to meet the requirements over time.

<16> Write phases of spiral model.

:-> The spiral model is similar to the incremental development for a system, with more emphasis placed on risk analysis. The spiral model has four phases: Planning, Design, Construct and Evaluation. A software project repeatedly passes through these phases in iterations (called Spirals in this model).

**Each cycle in the spiral is divided into four parts:**

1. **Objective setting:** Each cycle in the spiral starts with the identification of purpose for that cycle, the various alternatives that are possible for achieving the targets, and the constraints that exists.
2. **Risk Assessment and reduction:** The next phase in the cycle is to calculate these various alternatives based on the goals and constraints. The focus of evaluation in this stage is located on the risk perception for the project.
3. **Development and validation:** The next phase is to develop strategies that resolve uncertainties and risks. This process may include activities such as benchmarking, simulation, and prototyping.
4. **Planning:** Finally, the next step is planned. The project is reviewed, and a choice made whether to continue with a further period of the spiral. If it is determined to keep, plans are drawn up for the next step of the project.

<17> Write agile manifesto principles.

:-> The four core values of Agile software development as stated in the Agile Manifesto are as follows:

1. Individuals and interactions over processes and tools.
2. Working software over comprehensive documentation.
3. Customer collaboration over contract negotiation.
4. Responding to change over following a project plan.

## The 12 principles

The Agile Manifesto goes beyond scoping out a software project and then developing and releasing it. Perhaps the most important aspect of the manifesto is the close coordination and regular interactions with those who will be using the new or updated software.

The following are the 12 principles of the Agile Manifesto:

* Meeting end users' needs with early and [continuous delivery](https://www.techtarget.com/searchitoperations/definition/continuous-delivery-CD) of work.
* Being open to changes in requirements even late in the project.
* Delivering completed work at regular intervals, preferably short ones.
* Working with the project team and business owners daily.
* Assembling a motivated team, providing them with the right environment and support, and trusting them.
* Communicating face-to-face regularly.
* Using completed work to measure progress.
* Creating processes that promote sustainable efforts and a constant pace of work.
* Requiring continuous attention to excellence through good design.
* Encouraging simplicity.
* Recognizing that the best work emerges from self-organized teams that deliver the best architectures and designs.
* Reflecting regularly on how the team can be more effective and fine-tuning and adjusting the approach.

<18> Explain working methodology of agile model and also write pros and cons.

:-> Agile methodology is a project management framework that breaks projects down into several dynamic phases, commonly known as sprints. In this article, get a high-level overview of Agile project management, plus a few common frameworks to choose the right one for your team. Scrum, Kanban, waterfall, Agile.

**Cons of agile methodology**

**1. Transfer dificulties**

When switching from one management style to another for an entire department, it may take time to fully acclimate to the new responsibilities and style of how projects progress within the system.

**2.variable goals**

Because the agile methodology focuses on multiple goals at a time, certain goals may not receive as much focus throughout the entire project.

**3.Lake of documentation**

In the agile methodology system, documentation is less important than reactionary planning and progress.

**4.Less documented improvement**

Because agile methodology strategies focus on reactionary improvement rather than documented improvement phases, documented success and calculated methodology strategies may be ineffective throughout the project.

**5.Goal focus shifting**

Since agile methodology involves focus shifting based on which part of a project requires the most attention, it may be difficult to lead all team members toward a singular goal. In later stages of the project, it may become preferable to have the majority of department members focusing on a singular goal.

**6.Less predictability**

Because agile production relies on continuous improvement and customer feedback, the department may not be able to predict profits before production begins.

**Pros of agile methodology**

**1.Timely delivery**

Because it focuses on product placement, the agile strategy allows a department to deliver products to clients as quickly as possible.

**2.Adaptability**

Since the improvement increments between product delivery dates are small, projects can easily change and adapt when underneath an agile strategy system.

**3.Ease of collaboration**

Because agile working requires much feedback between the client and employees, agile methodology introduces a great system for collaborating between both customers and other departments.

**4.Increased performance improvement**

Because agile department test product increments as they produce them, department members can quickly react to problems as they may arise.

**5.Transparency**

With the agile approach to departmental work, both potential issues and process improvements become evident with each production cycle. This type of management allows employees to both correct mistakes and improve production details quickly.

**6.Continuous improvement**

Because the agile department strategy relies on improving a process as it produces products, improvement can directly effect the next product.

**7.Higher profits**

The agile department focuses on producing a continuously improving product rather than a perfect one. This enables agile departments to gain profits as quickly as possible, since each product profit brings feedback back to the team.

**8.Less preparatory work**

Because agile development focuses on the product more than process improvement, an agile department can produce a product more quickly than other management methods.

<19> Draw usecase on Online shopping product using COD

:->

<20> Draw usecase on Online shopping product using payment gateway.

:->