1). What is SRS?

SRS stands for Software Requirement specification is the complete description of software system that is developed as a product. This document will be useful right from first stage of development till the process is going on, even after deployment also this document will be helpful. When we have to do any changes in the software function, even extending the software project with new features this SRS will make DEV understand about the coding part and what changes they have to do with new features in software functions, even next team who will work on new features if new team got hired they get full information from this requirement specification. It includes all the use cases that describe all the information and interactions that will be useful for software users. Use case also means as functional requirements. Additionally, it also contains nonfunctional requirements or supplementary requirements. Nonfunctional requirements impose design constraints while implementation, there are performance requirements, quality standards. This SRS helps in enhancing the quality of a software even after the working model got deployed. For the maintenance part this will be a very useful tool. The sort of outline presented in SRS describes an appropriate approach taken along with the full content as per requirement.

For SRS there is a recommended approach that is IEEE 830- 1998, describing the content and qualities of good SRS and it presents a sort of outline of software as a sample.

2). What is OOPS?

Object Oriented Programming System.

Object means a real word entity, which exists in physical form in the universe such as, tangible things: Car printer, Interaction: contract, sale, Specification: Color, shapes, Roles like employee, boss etc.

Object Oriented is a methodology that we use to get the physical appearance of an object. It’s a program that uses methods or patterns or structure that results in designing software with the help of classes and objects. For example: A car is an object, its attributes are weight, color and its functions are driving, acceleration, speed. Object here is car and classes that presents the blueprint is it designing the attributes and functions and with help of object and classes we built that car and the physical appearance as a Car.

It’s a software programming methodology which we follow by using objects and classes. A model, pattern or paradigm used during designing a program.

Object oriented is an approach to software development that uses an application with the help of real-world objects models. For example, Employees, Car, Bank accounts etc. A class defines the properties and methods of a real-world object.

OOPS simplifies software development with the help of object orientation and software maintenance with the help of concepts.

OOP concepts are Object, Class, Inheritance, Polymorphism, Abstraction and Encapsulation.

3). Basic Concepts of OOPS?

Object Orientation programming system has 3 basic concepts or principle of OOP are:

1. Encapsulation: It’s a practice of including in an object everything it needs hidden from other objects. The internal state is usually not accessible by other objects.

* It’s a process of hiding information details and protecting data from the behavior of the object. For example: If we login Rediff mail and Gmail account, we get access to login create account password entering but the software which is running behind the login page, that is internal process, we will not get access to it.
* DEV uses encapsulation class that is easy for them to test during unit testing.
* Encapsulation is a way to restrict the direct access to some components of an object, so users cannot access the state values of variable of a particular object.
* It helps in placing the functions and data that work in the same place. While working with programming languages it is not very clear about which functions work with which variable but with the help of OOP, we got a clear framework to place relevant functions and data together in the same object.

1. Inheritance: It's an important concept in OOP. Inheritance means code reusability. The process of forming new class from the existing class, where existing class is the base class and new class that is formed by using base class is called as derived class. One class inherits the characteristics of another class is Inherits. We can also call this a relationship “IS A” that is inherits.

* Inheritance: is the feature that helps in reducing code size. In java Inheritance supports Single level as parent, multiple level as children. Grandparent-> Parent-> Children classes and interfaces. In class context, it also refers to interface inheritance.
* It’s a mechanism in which one class acquires the properties of another class. For example: a child inherits the traits from its parents with inheritance. We can reuse the fields and methods of existing class. Hence inheritance facilitates re usability and is an important concept of OOPs.

1. Polymorphism: Means “having many forms”.

* It allows different objects to respond to the same message in different ways, a response specific to the type of object.
* Poly means many, single function or an operator functioning in many ways differently upon the usage is called polymorphism. For example, message display Details () of the person class but when asked same to student () it will be different as per student object that is enrollment number.
* Most important aspect of an object is its behavior (things it can do). A behavior is initiated by sending message to the usual object by calling it as a method.
* There are two forms in Polymorphism, Overloading (compile time) and second is Over riding (Run time).

4). What is object?

An object represents an individual, incident, unit or entity, identifiable item whether it is real or that it is abstract with a well-defined role in the domain’s problem.

* An object responsibility is to know and responsibility to do. Object can be anything it may be tangible things such as Car, Printer. It can be Interactions such as Contract, Sale. It can be Roles such as Employee, boss. It can be Specification such as Colors, shape. It can be incidents such as flight, overflow.
* Basic unit of OOP = Object.
* Object = Data + Function.
* Object = Code + Methods.
* For example: A car is an object. Car has attributes such as weight, color and methods drive, brake, accelerate.

5). What is Class?

Class defines as a blueprint of an object. This doesn’t define any data, but it does define what class means, that is what an object of class consists of, and operations need to perform on such object.

* A class is like an object, constructor or blueprint for creating objects.
* A class represents an abstraction of the object and properties and behavior of that object.
* Class = Blueprint of object.
* Class = Defining object.
* Class = Template for an object.
* Class describing properties and behavior but without actual existence.
* Objects can be many for a class.
* Objects have their existence, but class has no existence.
* Example = car or laptop is an object. When class = blueprint is created then designing formed on initial stage then the actual car or laptop built based on that class. Here we do not buy blueprints but actual objects like cars or laptops.

There are 2 steps of OOP.

1. Making Classes: Creating, extending and reusing abstract data types.
2. Making Object interact: Creating objects from abstract data types and defining their relationships.

6). What is Encapsulation?

* Encapsulation is the fundamental of OOP that is bundling the data with the methods and operate on that data.
* In Java encapsulation is a process wrapping up data and behavior of an object in a single unit, this single unit is a class for property data and method behavioral.
* Encapsulation is used to hide the values or state of a structure data object inside a class, preventing unauthorized parties' direct access to them.

Main Advantage of Encapsulation

* Security of Data.
* Protects an object from unwanted access by clients.
* Allows access to a level without revealing the complex details about below that level.

Encapsulation binding the data with its related functionalities. Functionalist = methods + data. Functionality = Code + variable. Base is called class = variables and method in one place.

Java Encapsulation = hide (restrict access), to critical data members in your code which improves security.

If a data member is declared “private” then it can only access to same class. No outside class can access data member (variable) of that class.

If you need to access these variables you have to use “public”, “getter” and “setter” methods.

7). What is Inheritance?

Means that one class inherits the characteristics of another class. Also called “IS A” that is inheritance.

One of the useful aspects of OOP is reusability of a code and helps to reduce code size. NAME suggest inheritance is the process of forming new class from an existing class, existing is base class and another class that is new class or derived class.

* Example: A car is a vehicle. Vehicle -> is base class, parent class, super class. And car takes something from base class that is vehicle and some specific from car functions, then CAR is derived class, sub class, children class.

Reusability facility is an important concept in OOPs. Types of inheritance: Single inheritance (one class only) Class A -> Class B. That is Super Class -> Sub Class.

* For example: A Dog is an animal. Here animal is a base class and Dog is a derived class. Dogs are also called sub class and animal called super class. Dog is called child class and animal called parent class.
* Means Dog is called as Derived class, sub class and child class and animal are called as base class, parent class and super class.

One of the useful aspects of OOP is code re-usability. Name suggest inheritance is the process of forming a new class from an existing class called as base class where new class is formed called as inherit as derived class.

In java supports single parent, multiple child inheritance and multi-level inheritance (Grandparent -> Parent -> Child -> for classes and interfaces.

* Class context = inheritance is referred to as implementation.
* Class context = interface context also refers to as inheritance.

8). What is Polymorphism?

Most important concept in OOP is Polymorphism. In simple terms Poly means having many forms. The ability to change form is polymorphism. It allows different objects to respond to the same message in their own behavior.

* Polymorphism = Single function operating in many ways.
* Polymorphism = operator performing as per the ability of changing form.
* For example: Goal is communication and if you have smart phone then the mode you select is up to you on how to communicate. This mode can be a call, text, video call, audio call. Here Goal of communication can be solved in various ways as approach is different is called polymorphism.
* Two types of Polymorphisms are there: -

1. Compile time (overloading).
2. Running time (over riding).

Defining overloading when existing function or operator is made to operate on new data type, is said to be overloaded. In this method the class appears to have the same name but has a different signature.

For example: If we take the example of bank account. Then Account can be Parent class and its function that is deposit and withdraw can be child class. Here parent class having same function so there is no need to define it separately.

* Withdraw method for saving account is also called privilege class, when extending into check balance and then withdraw class.
* But when withdraw class uses check bank balance and then withdraw class is said to be privilege method or privilege class that is over riding.
* As custom withdraw action takes place is called privilege class and overdraft facility.

Defining overriding we will be able to override the methods of the base class so that meaningful implementation of the base class method can be derived class.

In java signature of method changes that include parameters type, order of parameters.

Overriding = can widely extend accessibility. But not narrow: if it is private in base class then child class can make it public but not vice versa. Example; Doctor -> works at hospital -> Treat Patient () -> then Treat Patient () -> Incision () Surgeon. Then in this first is Treat Patient () then overriding into Incision () surgeon.