MODULE: 1 (SDLC)

**1) What is software? What is software engineering?**

* Software is a set of instructions, data or programs used to operate computers and execute specific tasks.

**2)Explain types of software**

* Application software. The most common type of software, application software is a computer software package that performs a specific function for a user, or in some cases, for another application. An application can be self-contained, or it can be a group of programs that run the application for the user. Examples of [modern applications](https://www.techtarget.com/searchcio/feature/The-rise-of-modern-applications-Why-you-need-them) include office suites, graphics software, databases and database management programs, web browsers, word processors, software development tools, image editors and communication platforms.
* System software. These software programs are designed to run a computer's application programs and hardware. System software coordinates the activities and functions of the hardware and software. In addition, it controls the operations of the computer hardware and provides an environment or platform for all the other types of software to work in. The OS is the best example of system software; it manages all the other computer programs. Other examples of system software include the [firmware](https://www.techtarget.com/whatis/definition/firmware), computer language translators and system [utilities](https://www.techtarget.com/whatis/definition/utility).
* Driver software. Also known as device drivers, this software is often considered a type of system software. Device drivers control the devices and peripherals connected to a computer, enabling them to perform their specific tasks. Every device that is connected to a computer needs at least one device driver to function. Examples include software that comes with any nonstandard hardware, including special game controllers, as well as the software that enables standard hardware, such as USB storage devices, keyboards, headphones and printers.
* Middleware. The term middleware describes software that mediates between application and system software or between two different kinds of application software. For example, middleware enables Microsoft Windows to talk to Excel and Word. It is also used to send a remote work request from an application in a computer that has one kind of OS, to an application in a computer with a different OS. It also enables newer applications to work with legacy ones.
* Programming software. Computer programmers use programming software to write code. Programming software and programming tools enable developers to develop, write, test and [debug](https://www.techtarget.com/searchsoftwarequality/definition/debugging) other software programs. Examples of programming software include assemblers, compilers, debuggers and interpreters.

**3)What is SDLC? Explain each phase of SDLC**

* SDLC or the Software Development Life Cycle is a process that produces software with the highest quality and lowest cost in the shortest time possible. SDLC provides a well-structured flow of phases that help an organization to quickly produce high-quality software which is well-tested and ready for production use.

1. Planning

The planning phase involves project and product management. This may include:

Resource allocation (both human and materials)

Capacity planning

Project scheduling

Cost estimation

Provisioning

Ideally, Project Managers and Development staff collaborate with Operations and Security teams to ensure all perspectives are represented.

The outcomes of the planning phase includes:

Project plans

Schedules

Cost estimates

Procurement requirements

2. Requirements

* The business must communicate with tech teams to convey their requirements for new development and enhancement. The requirements phase gathers these parameters from business stakeholders and Subject Matter Experts (SMEs.)

Architects, Development teams, and Product Managers work with the SMEs to document the business processes that need to be automated through software. The output of this phase in a Waterfall project is usually a document that lists these requirements. Agile methods, by contrast, may produce a backlog of tasks to be performed.

3. Design and prototyping

* Once the requirements are understood, software architects and developers can begin to design the software. The design process uses established patterns for application architecture and software development. Architects may use an architecture framework such as [TOGAF](https://en.wikipedia.org/wiki/The_Open_Group_Architecture_Framework) to compose an application from existing components, promoting reuse and standardization.
* Developers use proven [Design Patterns](https://en.wikipedia.org/wiki/Design_Patterns) to solve algorithmic problems in a consistent way. This phase may also include some rapid prototyping, also known as a [spike](https://en.wikipedia.org/wiki/Spike_%28software_development%29), to compare solutions to find the best fit. The output of this phase includes:
* Design documents that list the patterns and components selected for the project
* Code produced by [spikes](https://www.scaledagileframework.com/spikes/), used as a starting point for development

4. Software development

* This phase involves the actual coding to produce the software itself. Depending on the methodology, this phase may be conducted in time-boxed “sprints,” (Agile) or may proceed as a single block of effort (Waterfall.)
* Regardless of methodology, the aim is for development teams to produce working software as quickly as possible. Business stakeholders should be engaged regularly, to ensure that their expectations are being met. The output of this phase is testable, functional software.

5. Testing

* The testing phase of the SDLC is one of the most important. It is impossible to deliver quality software without testing. There is a wide variety of testing necessary to measure quality:
* Code quality
* Unit testing (functional tests)
* Integration testing
* [Performance testing](https://raygun.com/platform/apm)
* Security testing
* The best way to ensure that tests are run regularly, and never skipped for expediency, is to automate them. Tests can be automated using Continuous Integration tools, [like Codeship](https://codeship.com). The output of the testing phase is improved software, ready for deployment to a production environment.

6. Deployment

* The deployment phase is, ideally, highly automated. In high-maturity enterprises, this phase is almost invisible; software is deployed the instant it is ready.
* For enterprises with lower maturity, or in some highly regulated industries, the process involves some manual approvals. However, even in those cases it is best for the deployment itself to be fully automated in a [continuous deployment](https://raygun.com/blog/continuous-deployment/) model. [Application Release Automation (ARA)](https://en.wikipedia.org/wiki/Application_release_automation) tools are used in medium and large-size enterprises to automate the deployment of applications to Production environments. ARA systems are usually integrated with Continuous Integration tools. The output of this phase is the release to Production of working software.

7. Operations and maintenance

* The operations and maintenance phase is the “end of the beginning,” so to speak. The Software Development Life Cycle doesn’t end here. Software [must be monitored](https://raygun.com/platform) constantly to ensure proper operation. Bugs and defects discovered in Production must be reported and responded to, which often feeds work back into the process. Bug fixes may not flow through the entire cycle, however, at least an abbreviated process is necessary to ensure that the fix does not introduce other problems (known as a [regression](https://en.wikipedia.org/wiki/Regression_testing).)

**4) What is DFD? Create a DFD diagram on Flipkart**

* A data flow diagram (DFD) is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement.

• What is Flow chart? Create a flowchart to make addition of two numbers

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task.

• What is Use case Diagram? Create a use-case on bill payment on paytm

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well.

**5)What is Flow chart? Create a flowchart to make addition of two numbers  
6)What is Use case Diagram? Create a use-case on bill payment on paytm.**