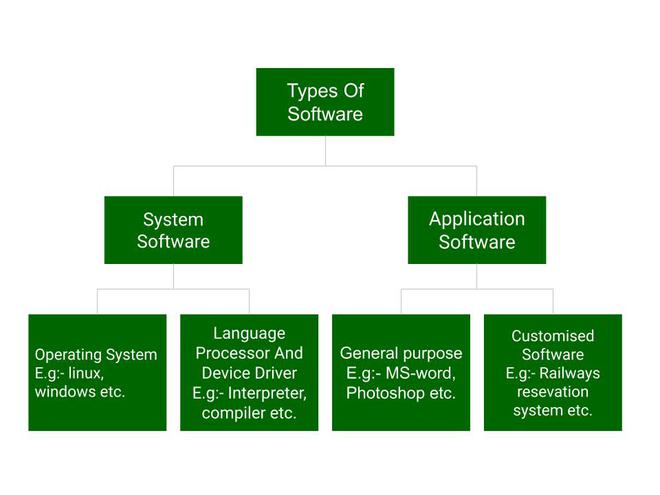
**1)What is software? What is software engineering?**

Software is a set of instructions, data or programs used to operate computers and execute specific tasks. It is the opposite of hardware, which describes the physical aspects of a computer. Software is a generic term used to refer to applications, scripts and programs that run on a device. It can be thought of as the variable part of a computer, while hardware is the invariable part.

Software Engineering is the process of designing, developing, testing, and maintaining software. It is a systematic and disciplined approach to software development that aims to create high-quality, reliable, and maintainable software. Software engineering includes a variety of techniques, tools, and methodologies, including requirements analysis, design, testing, and maintenance.

**2)Explain types of software.**

It is a collection of data that is given to the computer to complete a particular task. The chart below describes the types of software:



Above is the diagram of types of software. Now we will briefly describe each type and its subtypes:

1. **System Software**
   * Operating System
   * Language Processor
   * Device Driver
2. **Application Software**
   * General Purpose Software
   * Customize Software
   * Utility Software

**System Software**

[system software](https://www.geeksforgeeks.org/system-software/)is software that directly operates the computer hardware and provides the basic functionality to the users as well as to the other software to operate smoothly. Or in other words, system software basically controls a computer’s internal functioning and also controls hardware devices such as monitors, printers, and storage devices, etc. It is like an interface between hardware and user applications, it helps them to communicate with each other because hardware understands machine language(i.e. 1 or 0) whereas user applications are work in human-readable languages like English, Hindi, German, etc. so system software converts the human-readable language into machine language and vice versa.

**Types of System Software**

It has two subtypes which are:

1. **Operating System:** It is the main program of a computer system. When the computer system ON it is the first software that loads into the computer’s memory. Basically, it manages all the resources such as computer memory CPU, printer hard disk, etc., and provides an interface to the user, which helps the user to interact with the computer system. It also provides various services to other computer software. Examples of operating systems are Linux, Apple macOS, Microsoft windows, etc.
2. **Language Processor:**As we know that system software converts the human-readable language into a machine language and vice versa. So, the conversion is done by the language processor. It converts programs written in high-level languages like java,c,c++,python, etc(known as source code), into sets of instructions that are easily readable by machines(known as object code or machine code).
3. **Device Driver:**A device driver is a program or software that controls a device and helps that device to perform its functions. Every device like a printer, mouse, modem, etc. needs a driver to connect with the computer system eternally. So, when you connect a new device with your computer system, first you need to install the driver of that device so that your operating system knows how to control or manage that device.

**Application Software**

Software that performs special functions or provides functions that are much more than the basic operation of the computer is known as application system Or in other words, application software is designed to perform a specific task for end-users. It is a product or a program that is designed only to full fill end-users’ requirements. It includes word processors, spredsheet, database management, inventory, payroll programs, etc.

**Types of Application Software**

There are different types of application software and those are:

1. **General Purpose Software:**This type of application software is used for a variety of tasks and it is not limited to performing a specific task only. For example, MS-Word, MS-Excel, PowerPoint, etc.
2. **Customized Software:**This type of application software is used or designed to perform specific tasks or functions or designed for specific organizations. For example, railway reservation system, airline reservation system, invoice management system, etc.
3. **Utility Software:**This type of application software is used to support the computer infrastructure. It is designed to analyze, configure, optimize and maintains the system, and take care of its requirements as well. For example, [antivirus](https://www.geeksforgeeks.org/how-an-antivirus-works/), disk fragmenter, memory tester, disk repair, disk cleaners, registry cleaners, disk space analyzer, etc.

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

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. This methodology outlines a series of steps that divide the software development process into tasks you can assign, complete, and measure.

**->1. Planning & Analysis**

The first phase of the SDLC is the project planning stage where you are gathering business requirements from your client or stakeholders. This phase is when you evaluate the feasibility of creating the product, revenue potential, the cost of production, the needs of the end-users, etc. To properly decide what to make, what not to make, and what to make first, you can use a feature prioritization framework that takes into account the value of the software/update, the cost, the time it takes to build, and other factors. Once it is decided that the software project is in line with business and stakeholder goals, feasible to create, and addresses user needs, then you can move on to the next phase.

**->2. Define Requirements**

This phase is critical for converting the information gathered during the planning and analysis phase into clear requirements for the development team. This process guides the development of several important documents: a software requirement specification (SRS), a Use Case document, and a Requirement Traceability Matrix document.

**->3. Design**

The design phase is where you put pen to paper—so to speak. The original plan and vision are elaborated into a software design document (SDD) that includes the system design, programming language, templates, platform to use, and application security measures. This is also where you can flowchart how the software responds to user actions. In most cases, the design phase will include the development of a prototype model. Creating a pre-production version of the product can give the team the opportunity to visualize what the product will look like and make changes without having to go through the hassle of rewriting code.

**->4. Development**

The actual development phase is where the development team members divide the project into software modules and turn the software requirement into code that makes the product. This SDLC phase can take quite a lot of time and specialized development tools. It’s important to have a set timeline and milestones so the software developers understand the expectations and you can keep track of the progress in this stage. In some cases, the development stage can also merge with the testing stage where certain tests are run to ensure there are no critical bugs. Keep in mind, different types of product development software will have different specialties so you’ll want to pick the one that suits you best.

**->5. Testing**

Before getting the software product out the door to the production environment, it’s important to have your quality assurance team perform validation testing to make sure it is functioning properly and does what it’s meant to do. The testing process can also help hash out any major user experience issues and security issues. In some cases, software testing can be done in a simulated environment. Other simpler tests can also be automated. Performance testing: Assesses the software's speed and scalability under different conditions Functional testing: Verifies that the software meets the requirements Security testing: Identifies potential vulnerabilities and weaknesses Unit-testing: Tests individual units or components of the software Usability testing: Evaluates the software's user interface and overall user experience Acceptance testing: Also termed end-user testing, beta testing, application testing, or field testing, this is the final testing stage to test if the software product delivers on what it promises

**->6. Deployment**

During the deployment phase, your final product is delivered to your intended user. You can automate this process and schedule your deployment depending on the type. For example, if you are only deploying a feature update, you can do so with a small number of users (canary release). If you are creating brand-new software, you can learn more about the different stages of the software release life cycle (SRLC).

**->7. Maintenance**

The maintenance phase is the final stage of the SDLC if you’re following the waterfall structure of the software development process. However, the industry is moving towards a more agile software development approach where maintenance is only a stage for further improvement. In the maintenance stage, users may find bugs and errors that were missed in the earlier testing phase. These bugs need to be fixed for better user experience and retention. In some cases, these can lead to going back to the first step of the software development life cycle. The SDLC phases can also restart for any new features you may want to add in your next release/update.

**4)what is DFD? Create a DFD diagram on Flipkart.**

A **data-flow diagram** is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

There are several notations for displaying data-flow diagrams. The notation presented above was described in 1979 by Tom Demarco as part of structured analysis.

For each data flow, at least one of the endpoints (source and / or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes.

The data-flow diagram is a tool that is part of structured analysis and data modeling. When using UML, the activity diagram typically takes over the role of the data-flow diagram. A special form of data-flow plan is a site-oriented data-flow plan.

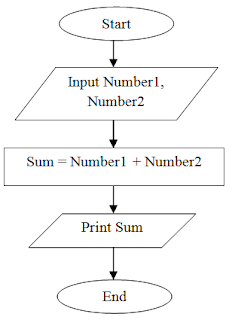
Data-flow diagrams can be regarded as inverted petri nets, because places in such networks correspond to the semantics of data memories. Analogously, the semantics of transitions from Petri nets and data flows and functions from data-flow diagrams should be considered equivalent.



**5)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 algorithm, a step-by-step approach to solving a task.

The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.



**6)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. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

