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. It is the opposite of hardware, which describes the physical aspects of a computer.

➢ The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

2. Explain types of software?

➢ There are different types of software that can run on a computer:

➢ system software, utility software, and application software.

➢ System Software: If you think of software as being in layers, the system software is the bottom layer: it sits between the hardware and the application software. Operating systems like Windows, macOS, Android and iOS are examples of system software. Operating systems are loaded into RAM when the device starts up, and have access to the hard drive.

➢ Application Software: This is everything else! Anything that is not an operating system, or a utility is an application or app. So, a word processor, spreadsheet, web browser, and graphics software are all examples of application software, and they can do many specific tasks. You can remove and add applications on your computer using the operating system. Application software like a word processor regularly directs the operating system to load and save files from and to the hard drive. When you are working on a file, it is saved temporarily in the RAM. It is only when you choose to save it that it is written to the hard drive.

➢ Utility software: Utility software is part of the system software and performs specific tasks to keep the computer running. Utility software is always running in the background. Examples of utility software are security and optimization programs. Security programs include anti-virus software that scans and removes viruses. Most computers will include some sort of anti-virus software, but you can add your own. Optimization programs can include tools for system clean-up, disk defragmentation, and file compression. These tools are typically installed as part of the operating system. They have access to the hard drive to keep it tidy.

3. What is SDLC? Explain each phase of SDLC.

➢ Software Development Life Cycle is the application of standard business practices to building software applications. It's typically divided into six to eight steps: Planning, Requirements, Design, Build, Document, Test, Deploy, Maintain. Some project managers will combine, split, or omit steps, depending on the project's scope. These are the core components recommended for all software development projects.

➢ Seven phases of SDLC:

3.1. **Planning:** In the Planning phase, project leaders evaluate the terms of the project. This includes calculating labor and material costs, creating a timetable with target goals, and creating the project's teams and leadership structure.

3.2. **Define Requirements**: Defining requirements is considered part of planning to determine what the application is supposed to do and its requirements. For example, a social media application would require the ability to connect with a friend. An inventory program might require a search feature.

3.3. **Design and Prototyping:** The Design phase models the way a software application will work. Some aspects of the design include:

Architecture – Specifies programming language, industry practices, overall design, and use of any templates or boilerplate.
User Interface – Defines the ways customers interact with the software, and how the software responds to input.
Platforms – Defines the platforms on which the software will run, such as Apple, Android, Windows version, Linux, or even gaming consoles.
Programming – Not just the programming language, but including methods of solving problems and performing tasks in the application.
Communications – Defines the methods that the application can communicate with other assets, such as a central server or other instances of the application.
Security – Defines the measures taken to secure the application, and may include SSL traffic encryption, password protection, and secure storage of user credentials.

3.4. **Software Development:** This is the actual writing of the program. A small project might be written by a single developer, while a large project might be broken up and worked by several teams. Use an Access Control or Source Code Management application in this phase. These systems help developers track changes to the code. They also help ensure compatibility between different team projects and to make sure target goals are being met. The coding process includes many other tasks. Many developers need to brush up on skills or work as a team. Finding and fixing errors and glitches is critical. Tasks often hold up the development process, such as waiting for test results or compiling code so an application can run. SDLC can anticipate these delays so that developers can be tasked with other duties. as waiting for test results or compiling code so an application can run. SDLC can anticipate these delays so that developers can be tasked with other duties.

3.5. **Testing:** It's critical to test an application before making it available to users. Much of the testing can be automated, like security testing. Other testing can only be done in a specific environment – consider creating a simulated production environment for complex deployments. Testing should ensure that each function works correctly. Different parts of the application should also be tested to work seamlessly together—performance test, to reduce any hangs or lags in processing. The testing phase helps reduce the number of bugs and glitches that users encounter. This leads to higher user satisfaction and a better usage rate.

3.6.	Deployment: In the deployment phase, the application is made available to users. Many companies prefer to automate the deployment phase. This can be as simple as a payment portal and download link on the company website. It could also be downloading an application on a smartphone. Deployment can also be complex. Upgrading a company-wide database to a newly developed application is one example. Because there are several other systems used by the database, integrating the upgrade can take more time and effort.

3.7.	Operations and Maintenance: At this point, the development cycle is almost finished. The application is done and being used in the field. The Operation and Maintenance phase is still important, though. In this phase, users discover bugs that weren't found during testing. These errors need to be resolved, which can spawn new development cycles. In addition to bug fixes, models like Iterative development plan additional features in future releases. For each new release, a new Development Cycle can be launched.
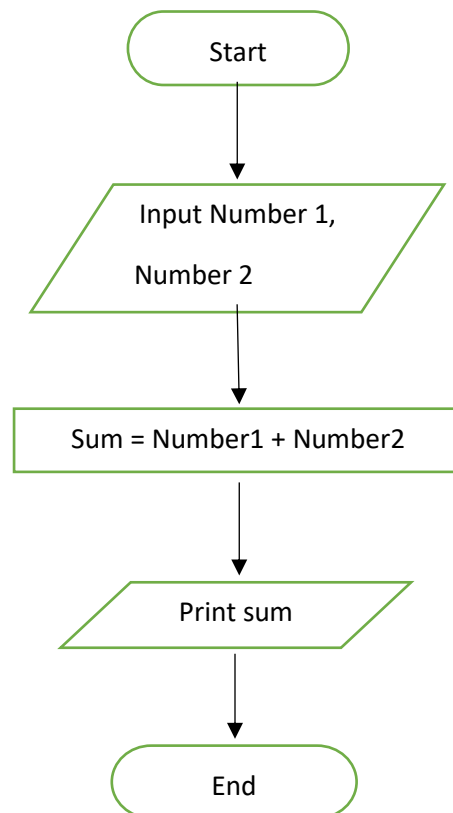
4. What is DFD? Create a DFD diagram on Flipkart.

➤ A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both. It shows how data enters and leaves the system, what changes the information, and where data is stored.

➤ DFD diagram on Flipkart:
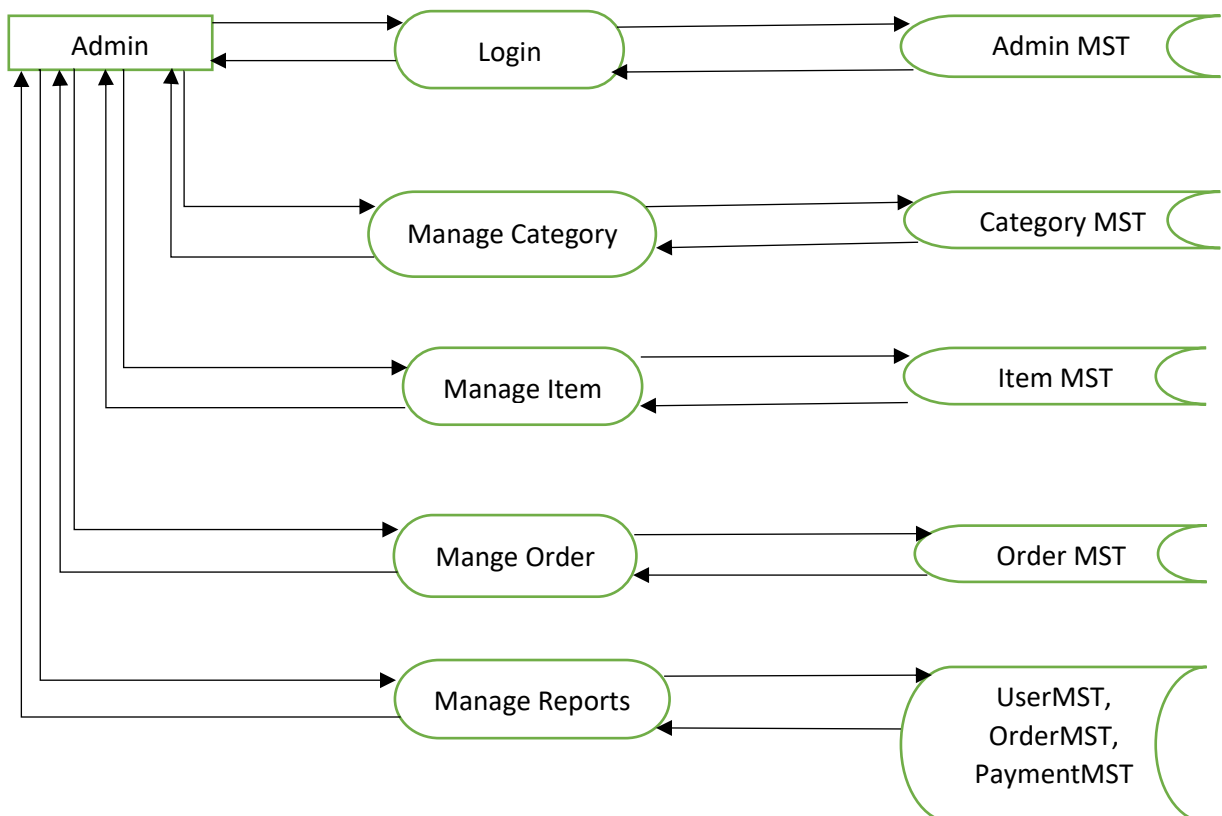
5. What is Flowchart? Create a flowchart to make addition of two numbers.

➢ A flowchart is a formalized graphic representation of a logic sequence, work or manufacturing process, organization chart, or similar formalized structure. The purpose of a flow chart is to provide people with a common language or reference point when dealing with a project or process.

➢ Flowchart tto make addition of two numbers:

```
      ( Start )
          |
          v
   / Input Number 1,  /
   / Number 2         /
          |
          v
   [ Sum = Number1 + Number2 ]
          |
          v
   / Print sum /
          |
          v
      ( End )
```

6. What is Use Case Diagram? Create a use-case on bill payment on Paytm.

➢ Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally.

➢ Use-case diagram on bille payment on Paytm:

```
        ( Install App )
             |
             v
      [ Start the App ]
             |
             v
        [ Login ]
             |
             v
    [ Select the Product ]
             |
             v
       [ Add to Cart ]
             |
             v
          [ Buy ]
             |
             v
        / Bill    /
       / Payment /
             |
             v
         ( Exit )
```