***SE - Assignment***

**Module – 1: - Overview of IT Industry**

**Q1.) What is Software? What is software engineering?**

**Software : -**

**-** Software refers to a set of instructions, programs, or data used to operate computers and execute specific tasks. It encompasses everything from operating systems and applications to games and utilities. In essence, software tells hardware (like your computer or smartphone) what to do and how to do it. There are two main types of software: system software, which manages the hardware and provides a platform for running applications, and application software, which fulfils specific user needs or tasks, like word processing, image editing, or browsing the internet.

**-** Think of software like a recipe. When you want to make a cake, you follow a recipe that tells you what ingredients to use and how to mix them together. In the same way, software is like a recipe for computers. It's a set of instructions that tells your computer what to do and how to do it. Just like a recipe helps you bake a cake, software helps your computer perform tasks like browsing the internet, editing documents, or playing games. There are different types of software for different purposes, just like there are different recipes for different dishes.

**Software Engineering : -**

**-** Software engineering refers to the disciplined approach of designing, developing, testing, deploying, and maintaining software applications or systems. It involves using various methodologies, tools, and techniques to create high-quality software that meets specific requirements and satisfies user needs. Software engineers utilize programming languages, frameworks, version control systems, and other technologies to build scalable, efficient, and reliable software solutions. The goal of software engineering is to produce software products that are robust, secure, and maintainable, while also considering factors such as performance, usability, and cost-effectiveness.

**-** Alright, imagine you want to build a treehouse. Software engineering is like planning, building, and maintaining that treehouse, but instead of wood and nails, you're using code and computers.

First, you'd figure out what you want the treehouse to do - like how many rooms it'll have and what cool features it'll have. That's like gathering requirements in software engineering.

Then, you'd sketch out a design for the treehouse, deciding where each room will go and how they'll connect. In software engineering, this is like designing how the different parts of the software will work together.

Next comes the fun part - building! You'd follow your design to actually construct the treehouse, putting in the walls, floors, and all the other bits. In software engineering, this is writing the code that makes the software do what it's supposed to do.

After the treehouse is built, you'd test it out to make sure everything works the way it should. Similarly, in software engineering, you test the software to find and fix any bugs or problems.

Finally, just like you'd do some upkeep on the treehouse over time to keep it safe and working well, in software engineering, you'd do things like update the software to add new features or fix any issues that come up.

**Q2.) Explain the types of software.**

**1. Application Software : -**

**-** Application software refers to computer programs or software designed to perform specific tasks or applications for users. Unlike system software, which manages the computer hardware and provides a platform for running applications, application software is developed to fulfil user needs directly. It includes a wide range of programs, such as word processors, spreadsheet applications, web browsers, multimedia editing tools, communication software, gaming software, and more. Application software enables users to accomplish various tasks on their computers or other digital devices, catering to different needs and preferences.

**-** Think of application software like tools in a toolbox. Each tool is designed for a specific job. For example:

A hammer is like word processing software (e.g., Microsoft Word), used for writing documents.

A wrench is like spreadsheet software (e.g., Microsoft Excel), used for organizing and calculating data.

A screwdriver is like web browsers (e.g., Google Chrome), used for accessing and browsing the internet.

A paintbrush is like multimedia software (e.g., Adobe Photoshop), used for editing images.

So, application software is like these tools, but for your computer. They help you do different things like write documents, browse the web, edit pictures, or play games.

**2. System Software (Inbuilt) : -**

**-** System software is a type of software that provides a platform for running applications and manages computer hardware resources. It acts as an interface between the hardware and the end-user applications. System software is essential for the proper functioning of a computer system.

**-** Think of system software as the backbone of your computer. It's like the behind-the-scenes manager that makes everything run smoothly so you can use your computer effectively. Ex. – OS System(Windows)

**-** System software is what keeps your computer running smoothly behind the scenes, handling all the important tasks so you can use your computer effectively.

**3. Driver software : -**

**-** Driver software, often referred to simply as "drivers," is a type of software that allows computer hardware devices to communicate with the operating system and other software applications. Essentially, drivers act as translators between the hardware devices (such as printers, graphics cards, network adapters, and peripherals) and the operating system.

**-** Imagineyour computer is a big city, and the operating system (like Windows or macOS) is the main language spoken there. Now, each hardware device in your computer is like a different neighbourhood with its own unique language. Without drivers, the operating system wouldn't be able to understand or communicate with these neighbourhoods.

That’s where driver software comes in. It's like a language translator that helps the operating system understand the language of each hardware device. When you install a new hardware device on your computer, you typically need to install the corresponding driver software so that the operating system can communicate with and control the device properly.

In summary, driver software is essential for enabling seamless communication between hardware devices and the operating system, ensuring that your computer hardware functions correctly and efficiently.

**3. Middleware software : -**

**-** Middleware software is a type of software that sits between different applications, systems, or components and facilitates communication and interaction between them. It acts as a bridge or intermediary, enabling seamless integration and interoperability between disparate software components or systems.

**-** Imagine you have two people who speak different languages and need to communicate with each other. Middleware software is like a translator who can understand both languages and facilitate communication between the two people. Similar in computer system : -

* You might have one software application running on one system and another application running on a different system, and they need to exchange data or interact with each other.
* You might have different components within a single application that need to communicate or share information.
* You might have legacy systems that need to be integrated with newer technologies or platforms.
* Middleware software provides a standardized way for these different systems or components to communicate and work together effectively. It often includes features such as messaging, data transformation, security, and transaction management.

**4. Programming Software : -**

**-** Programming software refers to tools and applications that programmers use to write, debug, test, and maintain computer programs and software applications. These software tools provide an environment and set of features to assist programmers in creating software efficiently and effectively.

- So, programming software is a collection of tools that programmers use to write code, make sure it works correctly, and keep track of changes over time.

**Q3.) What is SDLC? Explain each phase of SDLC*.***

**-** SDLC stands for Software Development Life Cycle. It's a structured process used by software development teams to design, develop, test, deploy, and maintain software applications or systems. The SDLC framework provides a systematic approach to software development, ensuring that projects are completed efficiently, on time, and within budget, while meeting quality standards and user requirements.

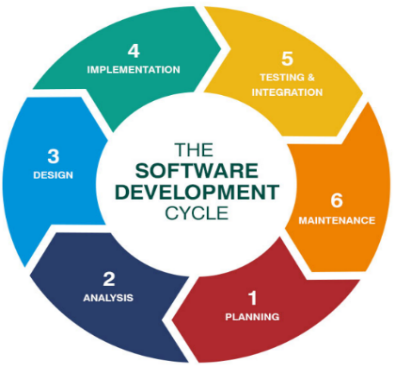
There are mainly some phases of SDLC : -

**1. Planning : -**

**-** Planning in the Software Development Life Cycle (SDLC) is like making a game plan before building something on the computer. It's about deciding what you want to create, figuring out if you have the right tools and people to do it, and making a schedule of how and when everything will get done. Just like planning a big project or a trip, it helps you stay organized, avoid problems, and reach your goal smoothly.

**2. Analysis : -**

**-** Analysis in the Software Development Life Cycle (SDLC) is like figuring out what you're going to build and why you're building it before you start. It's about understanding the problem, breaking it into smaller parts, writing down all the important details, exploring different ways to solve it, and getting the green light from the bosses before moving forward.

****

**SDLC [ Software Development Life Cycle ]**

**3. Design : -**

**-** In the Software Development Life Cycle (SDLC), the design phase is like drawing a map before going on a journey. It's about planning exactly how the software will function and what it will look like. This includes deciding on things like how users will interact with it, what tools and technologies will be used, and making sure it's both easy to use and secure. It's basically creating a detailed plan before starting to build the actual software.

**4**. **Implementation : -**

**-** In the Software Development Life Cycle (SDLC), implementation is like building a house after you've made the plans. It's when developers actually write the code and put together all the pieces to create the software. They test it along the way to make sure it works correctly, and they prepare it to be used by others once it's finished.

**5**. **Testing & Integration : -**

**-** Testing in the Software Development Life Cycle (SDLC) is like trying out a recipe before serving it to guests. It's about checking if the software works correctly, doesn't have any mistakes (bugs), and meets the needs of the users. Just like tasting the food to make sure it's delicious and safe to eat, testing ensures the software is reliable, performs well, and is user-friendly before it's used by people.

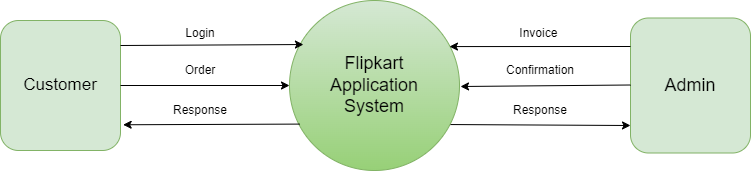
**6. Maintenance : -**

**-** Maintenance in the Software Development Life Cycle (SDLC) is like taking care of a pet or a plant. Once the software is built and being used, maintenance involves keeping it healthy and happy. This includes fixing any problems (like giving medicine to a sick pet), making updates to add new features or improve performance (like buying new toys or upgrading a plant's pot), and providing support to users if they have questions or need help (like answering questions about how to care for a pet or plant). Essentially, maintenance ensures the software stays in good shape and continues to meet users' needs over time.

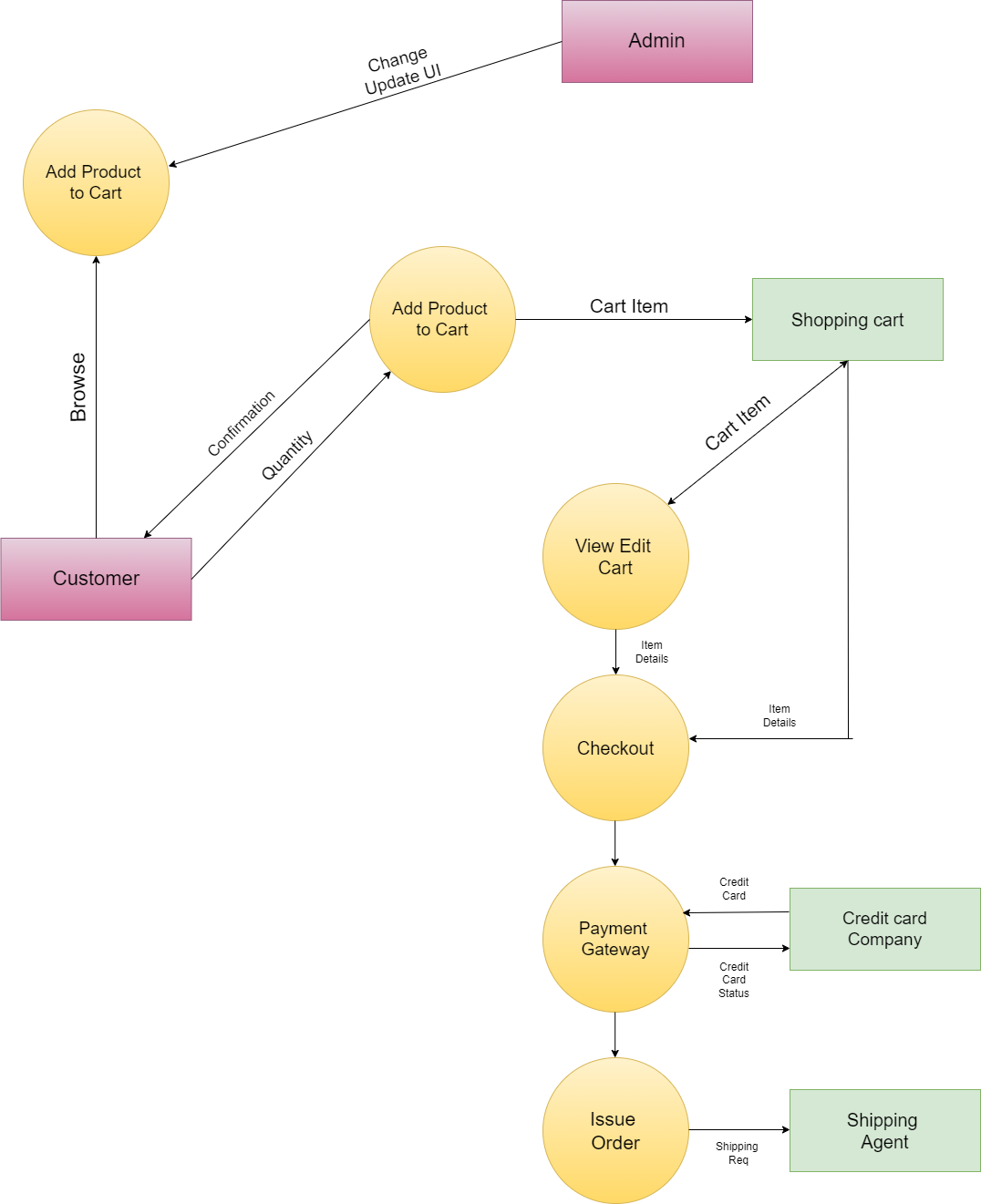
**Q4.) What is DFD? Create a DFD diagram on Flipkart.**

**-** Think of a Data Flow Diagram (DFD) like a map showing how information moves around in a system. Imagine you have a factory where raw materials come in, get turned into products, and then shipped out. In a DFD, you'd draw boxes to represent these steps, like "raw materials come in" and "products go out." Then, you'd draw arrows showing how the materials move from one step to another. It helps people understand how things flow through a system, making it easier to spot where things might get stuck or how to improve the process.

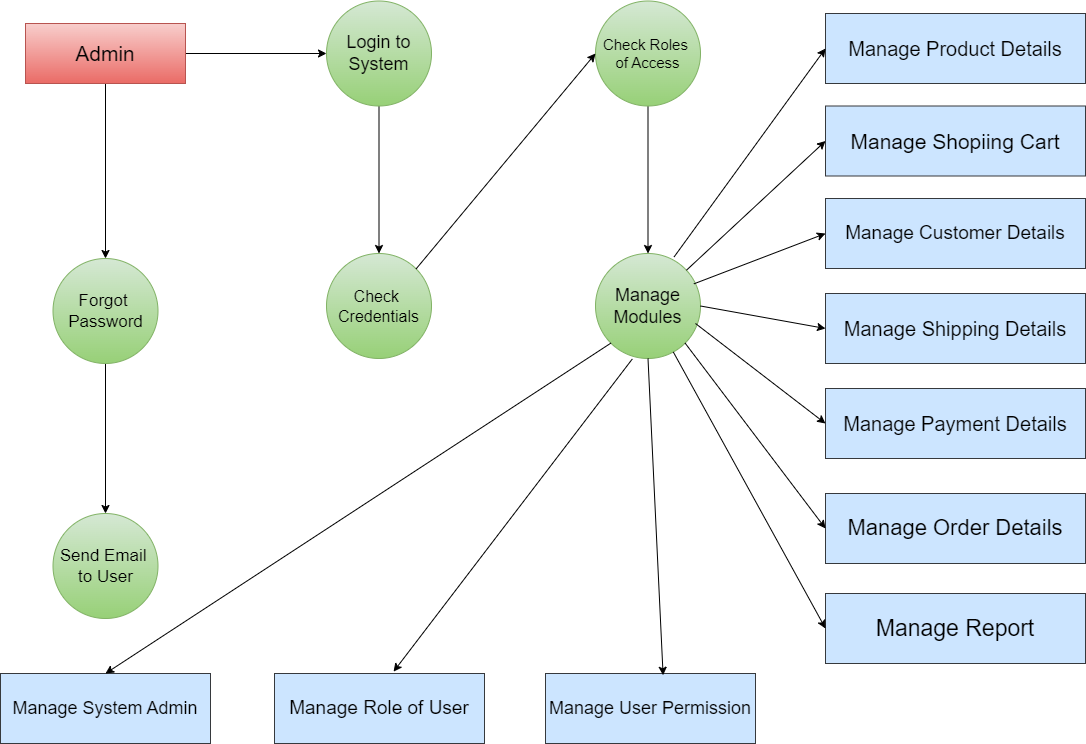
**Level 0 DFD**



**Level 1 DFD**



**Level 2 DFD**



**Q5.)** **What is Flow chart? Create a flowchart to make addition of two numbers.**

**-** A flowchart is a visual representation of a process or a series of steps. It uses shapes and arrows to show the sequence of actions or decisions that need to be taken to accomplish a task or solve a problem. It's like a map that helps you understand and follow a procedure from start to finish.

Imagine you're making a sandwich. A flowchart for making a sandwich would be like drawing little pictures or symbols to show each step, like getting bread, adding ingredients, and putting it all together. Arrows connect the steps to show the order you do things in. So, a flowchart is basically a picture guide that helps you see what you need to do next when you're following a process.

**Algorithm : -**

An algorithm is like a recipe or a set of instructions for solving a problem or completing a task. It's a series of steps that you follow in a specific order to achieve a desired outcome. Algorithms are used in computer science, mathematics, and everyday life to solve problems efficiently and effectively. They can be as simple as a recipe for baking cookies or as complex as the instructions for a computer program to sort data or play a game.

**Program : -**

#include<stdio.h>

Int main()

{

Int n1,n2,ans;

printf(“Enter 2 values : ”);

scanf(“%d%d”,&n1,&n2);

ans = n1 + n2;

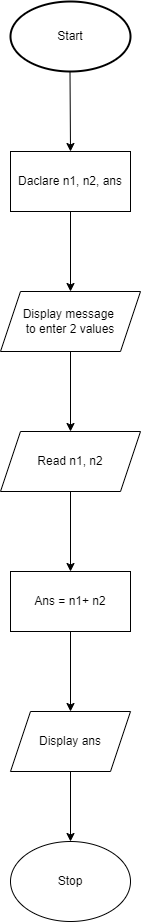
printf(“Your ans is : %d”,ans);

return 0;

}

**Algorithm : -**

1. Start
2. Declare n1,n2,ans
3. Display message to enter 2 variables
4. Read value of n1,n2
5. Ans -> n1 + n2
6. Display ans
7. Stop



**Q-6.)** **What is Use case Diagram? Create a use-case on bill payment on paytm.**

**-** A use case diagram is like a picture that shows who does what in a system or a situation. It uses simple drawings to show the different actions people or things can take and how they're connected. It helps us understand how everything works together.

**Paytm Use – case**

