**ASSIGNMENT – 1**

1. **What is Software? And Software Engineering?**

* Software is the language of computer.
* Is a collection of computer programs and related data that provide the instruction for the telling a computer what to do and how to do it.
* 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.

Types of software:

1) System software / operating system.

2) Application software

3) Programming software

Software Engineering - Software engineering is the art of developing quality software on time and within budget. Software engineering is a systematic approach to the design development operation and native view problem specification final program.

Software engineering is the field of computer science that deals with the design, development, testing, and maintenance of software applications. Software engineers apply engineering principles and knowledge of programming languages to build software solutions for end users.

- Software engineering is the multi-person construction of multi-version software

• Team-work

• Successful software systems must evolve or perish

• Change is the norm, not the exception

• Not constrained by physical laws

• limit = human mind

• balancing stake-holders

1. **Explain types of Software**

- There are following types of software

1) System software / operating system.

2) Application Software

3) Programming Software

1. **System Software:**

System software is a type of computer program that is designed to run a computer's hardware and application programs. If we think of the computer system as a layered model, the system software is the interface between the hardware and user applications. The operating system is the best-known example of system software. The OS manages all the other programs in a computer.

is the s/w used by the computer to translate inputs from various sources into a language which a machine can understand.

Example - Linux, window, macOS, Android, iOS

1. **Application Software:**

Application software is a kind of software that performs specific functions for the end user by interacting directly with it. The sole purpose of application software is to aid the user in doing specified tasks.

- Types of Application Software

1) Mobile app: - Application that run on mobile

- Ex. Instagram, Facebook, WhatsApp etc

2) Desktop app: - That run stand-alone in a desktop or laptop computer.

- Ex. Microsoft office suite which includes Word, Excel and PowerPoint.

- Ex. Outlook for email, and Firefox, Google Chrome, Mozilla are the web browser.

- Anti-virus is an application and so is the media player.

3) Web app: - That run on a web browser

- Ex. google.com, facebook.com, Amazon.in etc

1. **Programming Software:**

Programming software is a tool for creating computer code that allows computer software to operate. The computer technology field frequently uses overlapping terminology, which can be perplexing.

Is the process of designing, writing, testing, debugging, and maintaining the source code of computer programs.

Ex – Visual Studio, VS code, Sublime, Android Studio etc.

1. **Utility Software:**

Utility software provides additional functionality to enhance the performance of a computer system. Examples include antivirus software , backup software (Acronis True Image, Carbonite), and disk cleanup tools (CCleaner).

Ex - Norton, McAfee etc.

**5) Enterprise Software:**

These are software applications designed to meet the needs of organizations rather than individual users. Examples include Customer Relationship Management (CRM) software), Enterprise Resource Planning (ERP) software (SAP, Oracle ERP Cloud), and Supply Chain Management (SCM) software.

Ex - Oracle ERP Cloud, Microsoft Dynamics 365, Infor ERP etc.

**6) Embedded Software:**

Embedded software is specialized software that controls hardware devices and is typically built into the firmware of embedded systems. Examples include software in consumer electronics (smartphones, smart TVs), automotive systems (engine control units, infotainment systems), and industrial machinery.

Ex - Engine Control Unit (ECU) software, Flight Control system software etc.

**7) Artificial Intelligence (AI) Software:**

AI software utilizes algorithms and machine learning techniques to perform tasks that typically require human intelligence. Examples include virtual assistants (Siri, Google Assistant), recommendation systems (Netflix recommendation algorithm), and natural language processing (NLP) tools.

Ex – ChatGPT, Apple Siri, etc.

**3. What is Software Development Life Cycle (SDLC)?**

- The Software Development Life Cycle (SDLC) is a structured process used by software development teams to design, develop, test, and deploy high-quality software. It consists of several phases, each with its own objectives, deliverables, and activities. Here's a detailed explanation of each phase:

**1 - Requirement Gathering:**

* phase involves gathering requirements from stakeholders to understand what the software should do and how it should behave.
* Business analysts and stakeholders collaborate to define the scope, objectives, and functional/non-functional requirements of the software.
* Deliverables: Requirements Document, Functional Specifications Document.

**2 - Analysis:**

* The analysis phase sets the stage for the subsequent phases of the SDLC, including design, development, testing, deployment, and maintenance.
* By thoroughly understanding the requirements and constraints upfront, the development team can proceed with confidence, knowing that they are building a solution that addresses the needs of the stakeholders.

**3 - Design**:

* Design phase translates the requirements gathered in the first phase into a detailed technical blueprint.
* Architects and designers create system architecture, database design, user interface design, and other design documents.
* The design should be scalable, maintainable, and aligned with the requirements and constraints.
* Deliverables: System Architecture, Database Design, User Interface Design, Technical Design Document.

**4 - Implementation (Coding):**

* Development teams start coding based on the design documents produced in the previous phase.
* Developers follow coding standards, best practices, and use appropriate programming languages and frameworks.
* Code is typically version-controlled, and collaborative development tools may be used for better coordination.
* Deliverables: Source Code, Unit Tests.

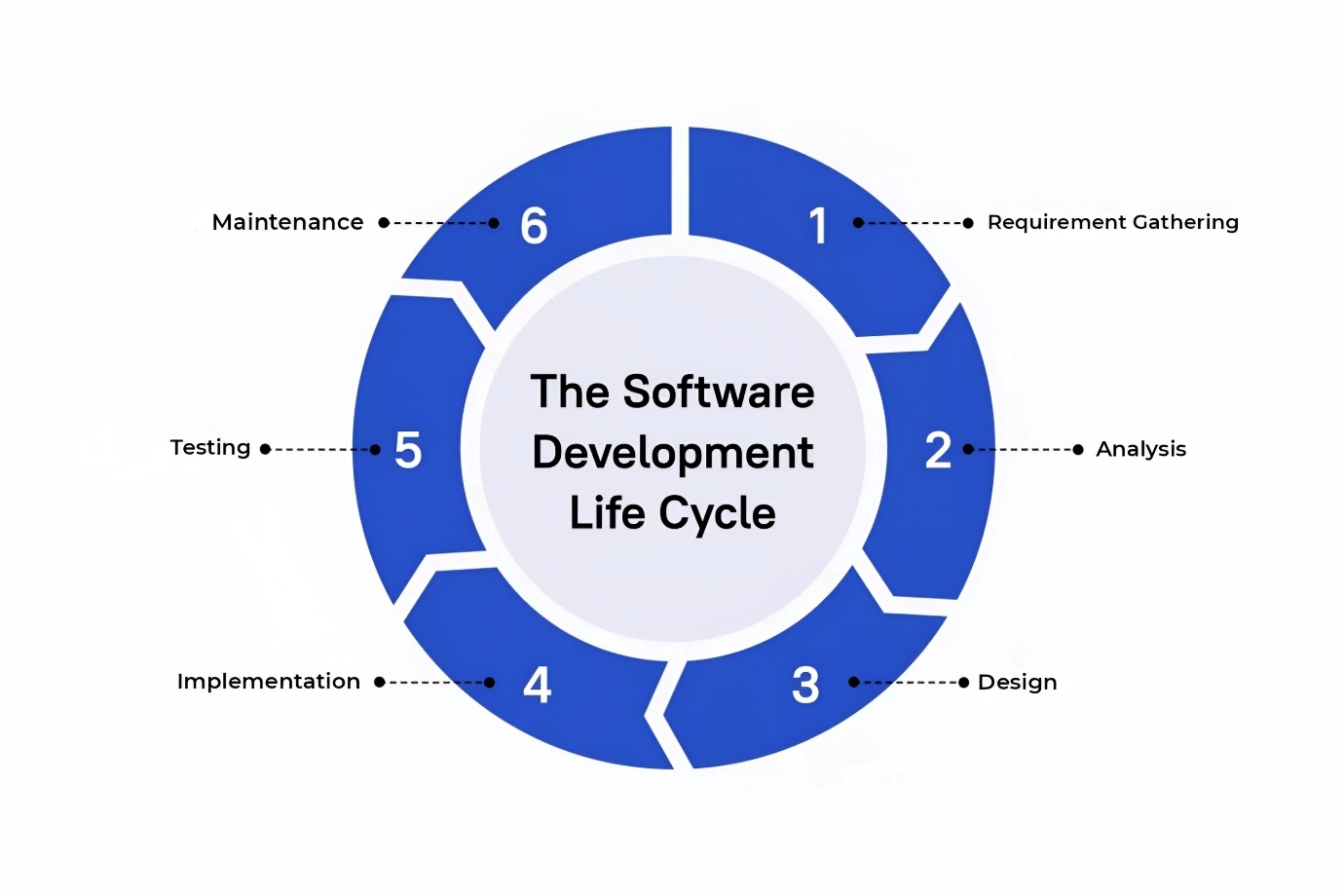
**5 - Testing:**

* This phase involves verifying and validating the software against the requirements.
* Testers create test cases based on requirements and execute them to find defects or deviations from expected behavior.
* Different types of testing are performed, including unit testing, integration testing, system testing, and user acceptance testing.
* Defects found during testing are reported, prioritized, and fixed by developers.
* Deliverables: Test Plan, Test Cases, Test Reports, Defect Reports.

**6 - Maintenance:**

* After deployment, the software enters the maintenance phase, where it is monitored, updated, and supported.
* Bug fixes, patches, enhancements, and upgrades are released based on user feedback and changing requirements.
* Support teams provide assistance to users, troubleshoot issues, and ensure the software operates as intended.
* Deliverables: Bug Fixes, Patches, Updates, Support Documentation.

**Software Development Life-cycle (SDLC)**



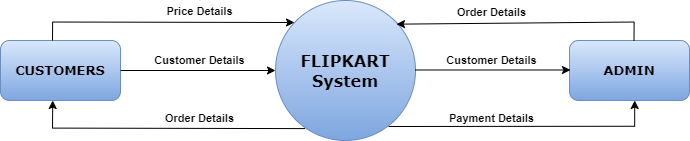
**4. What is DFD? Create a DFD diagram on Flipkart.**

- DFD stands for Data Flow Diagram. It's a graphical representation of how data flows through a system or a process. It's often used in software engineering and systems analysis to visualize the flow of data within a system.

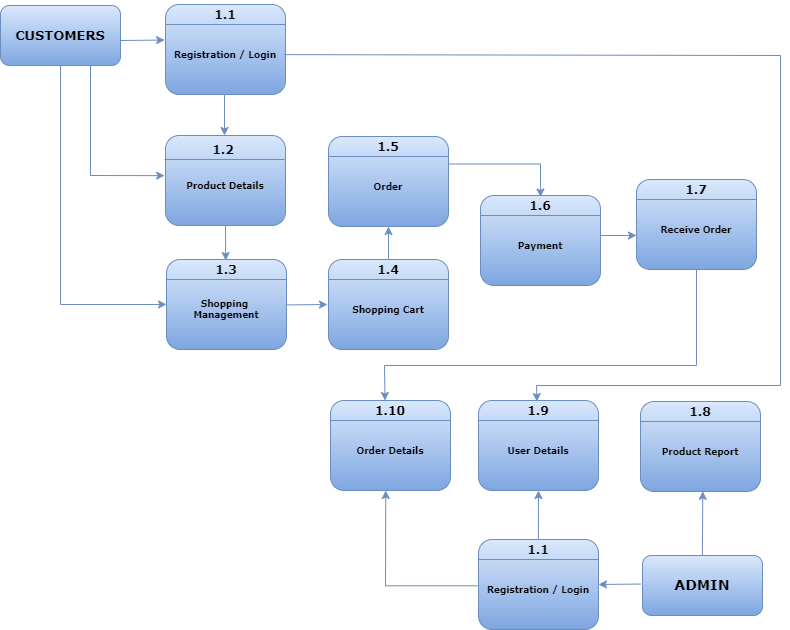
* **components of a DFD:**

1. **Processes**: These are the actions or transformations that take place within the system. Processes are represented by circles or rectangles in the diagram. Each process represents a specific function or action that manipulates data in some way.
2. **Data Flows**: These represent the movement of data between the various processes, data stores, and external entities in the system. Data flows are represented by arrows and indicate the direction in which data is moving.
3. **Data Stores**: These represent where data is stored within the system. Data stores are represented by rectangles with two parallel lines at the top and bottom. They can represent databases, files, or any other storage mechanism where data is kept.
4. **External Entities**: These represent the sources or destinations of data outside the system being modeled. External entities could be users, other systems, or anything else that interacts with the system but is not part of it. They are represented by squares or rectangles.

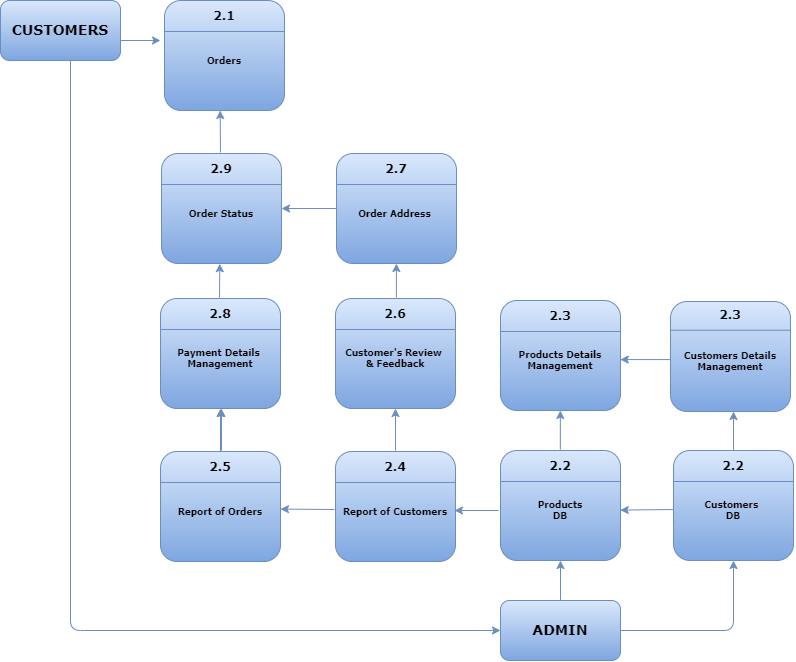
* **DFDs are typically drawn at different levels of detail:**
* **Level 0 DFD**: This is the highest level of abstraction, showing the entire system as a single process with external entities interacting with it.
* **Level 1 DFD**: This breaks down the processes of the level 0 DFD into more detailed subprocesses.
* **Level 2 DFD**: Further detail can be added with Level 2 DFDs, breaking down the processes of Level 1 into even smaller subprocesses.
* **DFD Diagram on Flipkart**
* **Level 0 - DFD:**

****

* **Level 1 - DFD:**

****

* **Level 2 - DFD:**

****

* **Explanation:**

1. **External Entities**:
   * **Customers**: Users who interact with the Flipkart platform to browse, search for products, make purchases, and leave reviews.
   * **Admin**: In the context of Flipkart, an "admin" would typically refer to an individual or a group of individuals who have administrative privileges or authority within the Flipkart system. This could include various responsibilities such as managing the platform's operations, overseeing user accounts, handling product listings, monitoring transactions, ensuring compliance with regulations, and addressing any technical issues that may arise.
2. **Processes**:
   * **Registration / Login**: Users provide their name, email, phone, and password on the Flipkart website or app, optionally verifying their contact information and login Returning users enter their registered email/phone and password to access their Flipkart account and start shopping.
   * **Products Details**: Customers can get details for specific products using keywords or filters.
   * **Shopping Management:** Shopping management on Flipkart involves users browsing products, adding items to their cart, and proceeding to checkout. Flipkart's system handles inventory management, order processing, payment transactions, and delivery logistics to ensure a smooth shopping experience for customers.
   * **Shopping Cart**: After finding a product they want to purchase, customers can add it to their shopping cart.
   * **Order**: Customers initiate the checkout process to place an order for the products in their cart.
   * **Payment**: Flipkart processes the payment for the order, including transaction validation and authorization.
   * **Order Details:** Admin of Flipkart System get all orders details.
   * **User Details:** Admin have all users details that who is active or not, there order and many more.
   * **Product Report:** Admin has all products report like there stock, price increment decrement, offer, sale of products etc.
   * **Admin Login:** Admin also have to register/login to access Flipkart's admin dashboard
3. **Data Stores**:
   * **Product Database**: Stores information about the products available on Flipkart, including details like name, description, price, and seller information.
   * **Customer Database**: Stores information about registered customers, including account details, order history, and payment information
4. **What is Flowchart? Flowchart to make addition of two numbers?**

- A flowchart is a graphical representation of a process or algorithm, typically using symbols connected by arrows to show the flow of control or data within the system. It's commonly used in various fields such as software engineering, business management, and education to visually illustrate the steps of a process or the logic of an algorithm. Flowcharts help in understanding complex systems, identifying bottlenecks, and designing efficient workflows.

**Components of a Flowchart:**

1. **Start/End Symbol**: Usually represented by an oval or rounded rectangle, this symbol marks the beginning and end of the process. It typically contains the word "Start" or "End."
2. **Process Symbol**: Represented by a rectangle, this symbol indicates an action or operation that occurs within the process. It contains a brief description of the action.
3. **Decision Symbol**: Shaped like a diamond, this symbol represents a decision point in the process where a question is asked, and the flow of the process branches based on the answer (usually yes/no or true/false).
4. **Input/Output Symbol**: Represented by a parallelogram, this symbol indicates input or output of data into/from the process. It typically contains a description of the input/output data.
5. **Connector Symbol**: Represented by a small circle, this symbol is used to connect different parts of the flowchart that are located on different pages or sections of a large flowchart.
6. **Flow Arrows**: Arrows connect the symbols and indicate the flow or direction of the process. They show the sequence in which the steps of the process are performed.

**Usage of Flowcharts:**

1. **Process Documentation**: Flowcharts are commonly used to document and communicate complex processes in various fields such as business, engineering, software development, and healthcare. They provide a clear visual representation of how a process works.
2. **Problem-Solving**: Flowcharts help in identifying inefficiencies, bottlenecks, or errors in a process. By visually mapping out the steps, it becomes easier to analyze and improve the process.
3. **Software Development**: Flowcharts are widely used in software development to design algorithms, plan program logic, and visualize the execution flow of a program.
4. **Training and Education**: Flowcharts are valuable tools for training new employees or students. They provide a structured overview of processes and procedures, making it easier for learners to understand and follow.
5. **Decision Making**: Flowcharts can be used to model decision-making processes, enabling stakeholders to visualize different outcomes based on various choices

* **Algorithm**

**1)** Start

**2)** Declare a variable no1, no2, ans

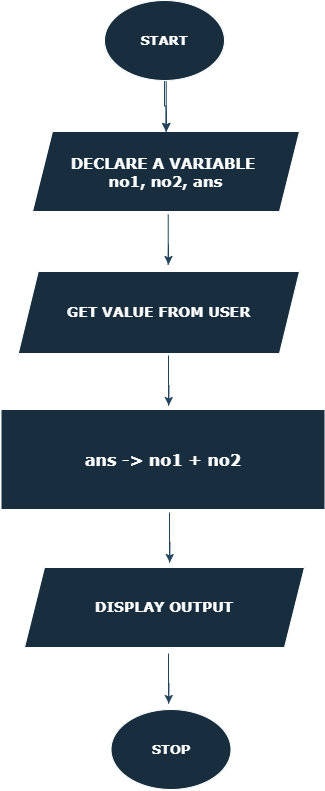
**3)** Get value from user

**4)** ans -> no1+no2

**5)** Display the Output

**6)** Stop

**Flowchart**

****

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

- A use case diagram is a graphical representation of interactions between users (actors) and system, depicting the various ways users interact with the system to achieve specific goals. It's a part of Unified Modeling Language (UML), commonly used in software engineering to illustrate the functional requirements of a system and its intended behavior from a user's perspective.

**Components of a Use Case Diagram:**

1. **Actor**: An actor represents a role or entity that interacts with the system. Actors can be users, external systems, or other software components. They are depicted as stick figures or labeled ovals outside the system boundary.
2. **Use Case**: A use case represents a specific functionality or task that the system provides to its users. Each use case describes a sequence of interactions between the system and one or more actors to achieve a particular goal. Use cases are depicted as ovals inside the system boundary and are labeled with descriptive names.
3. **Association**: An association is a relationship between an actor and a use case, indicating that the actor interacts with the system to perform the activities described by the use case. Associations are represented by solid lines connecting actors to use cases.
4. **System Boundary**: The system boundary represents the scope of the system being modeled. It encloses all the use cases and actors associated with the system.
5. **Include Relationship**: An include relationship indicates that one use case includes the functionality of another use case. It is depicted by a dashed line with an arrowhead pointing from the including use case to the included use case.
6. **Extend Relationship**: An extend relationship indicates that one use case may optionally extend another use case by adding additional behavior under certain conditions. It is depicted by a dashed line with an arrowhead pointing from the extending use case to the extended use case.

**Usage of Use Case Diagrams:**

1. **Requirements Analysis**: Use case diagrams help stakeholders understand the functional requirements of a system by visualizing how users interact with it to accomplish specific tasks or goals.
2. **Communication Tool**: Use case diagrams serve as a communication tool between stakeholders, developers, and designers, facilitating discussions about system functionality and user interactions.
3. **System Design**: Use case diagrams provide a foundation for system design by identifying key use cases and actors, which can then be further elaborated into detailed specifications, user interface designs, and system architecture.
4. **Testing**: Use case diagrams can inform test case development by identifying critical user interactions and system behaviours that need to be tested.
5. **Project Planning**: Use case diagrams help in project planning by providing a high-level overview of the system's functionality, which can inform resource allocation, scheduling, and prioritization of development tasks.

* **Use case Diagram on Bill Payment on Paytm**

