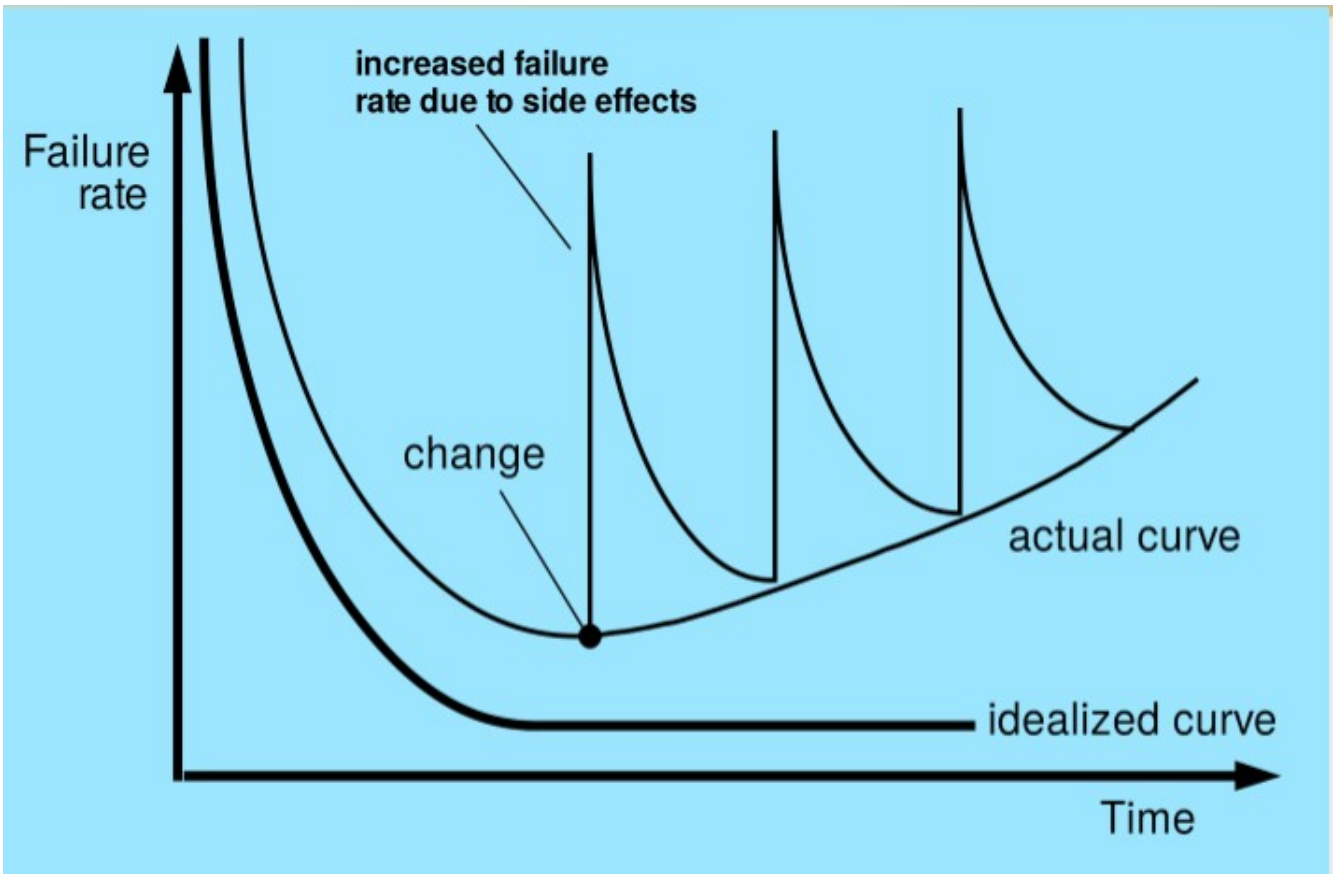
Introduction

1. Compare a Software with a program. Give suitable examples from both of them

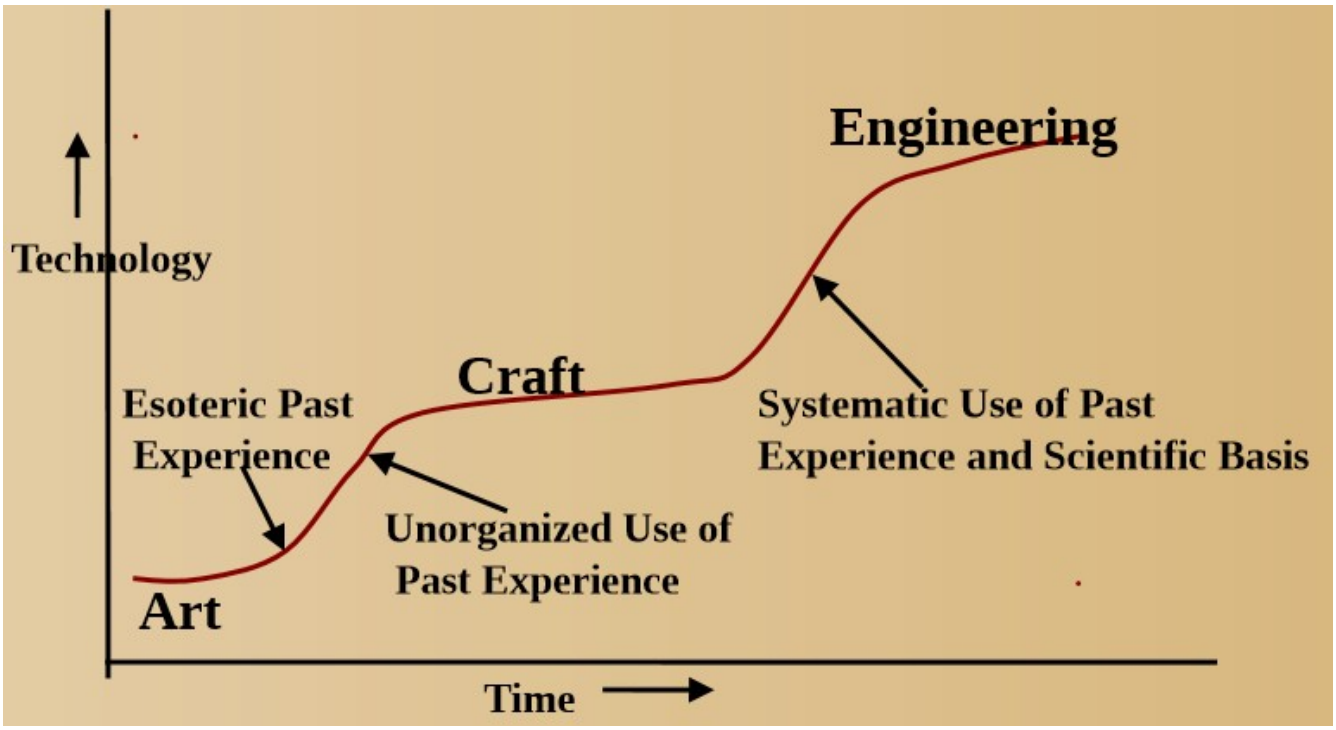
Program	Software
Usually small in size	Large
Author himself is sole user	Large number of users
Single developer	Team of developers
Lacks proper user interface	Well-designed interface
Lacks proper documentation	Well documented and user manual prepared
Ad-hoc development	Systematic development

Examples of Programs:
Examples of Software: Office, Chrome, etc

2. Draw and explain the Idealized and actual curves for software failure rate vs time.

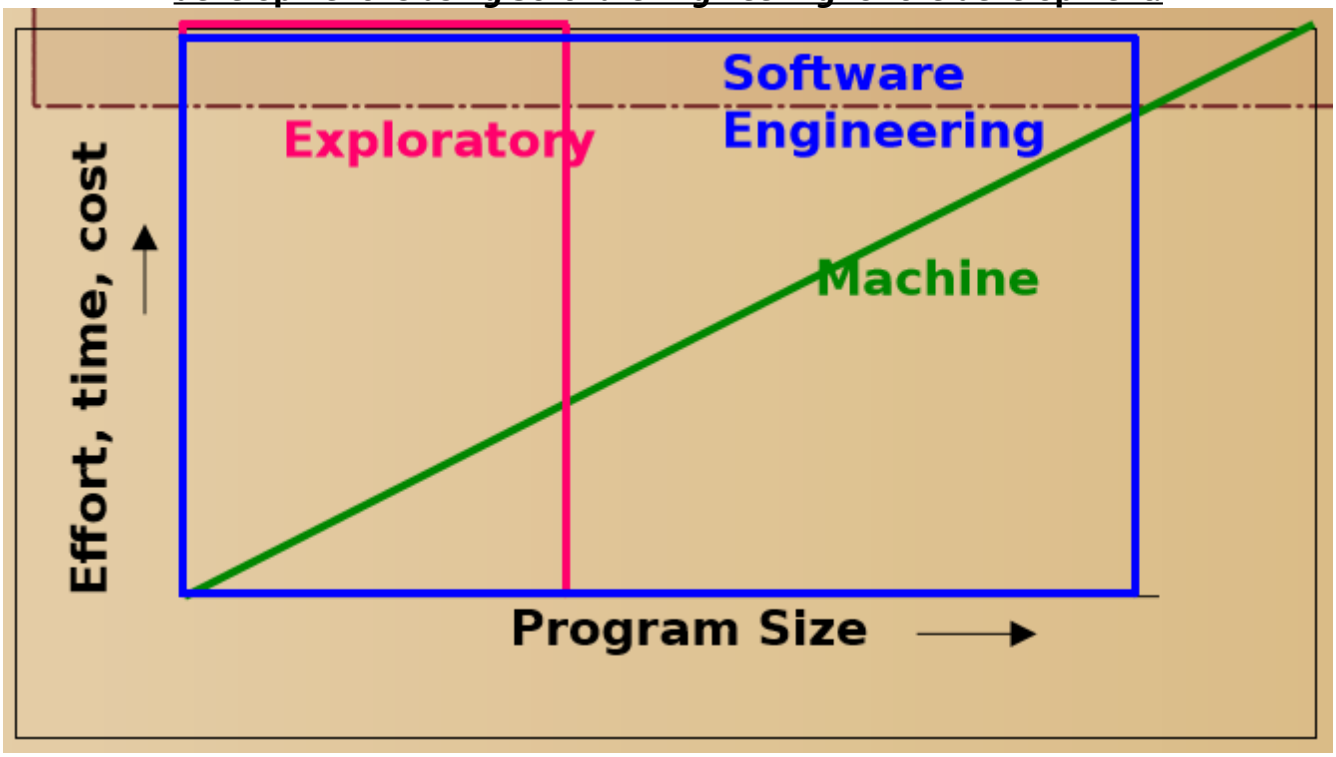


3. Draw and explain the evolution pattern of Technology



4. Define software quality. What quality assurance is needed for a system to manage the work schedule of nurses that respects all the constraints and regulations in force at a particular hospital? Justify your answer.

5. Compare the effort, time and cost requirements for exploratory style of software development vs using Software Engineering for the development.



6. Explain how abstraction and decomposition helps to overcome human cognitive limitations.

Abstraction: Simplify a problem by omitting unnecessary details. Focuses attention on one aspect of the problem.
Decomposition: Decompose a problem into many small independent parts. Each small part is easy to understand and can be easily solved.

7. What is the objective of conducting feasibility study in software development life cycle?

A feasibility study should be relatively cheap and quick. The result should inform the decision of whether or not to go ahead with a more detailed analysis.

8. What are the difference between generic software product development and custom software development?

Generic software product development	Custom software development
Specification is owned by the product developer	Specification is owned and controlled by the customer as the domain, requirement and environment are unique to the customer
The developer can quickly decide to change the specification in response to some external change	Changes have to be negotiated between the customer and the developer and may have contractual obligations

9. List five quality attributes of a good software.

Good software has the following attributes:
Usability: Users can learn it and fast and get their job done easily
Efficiency: It doesn't waste resources such as CPU time and memory
Reliability: It does what it is required to do without failing
Maintainability: It can be easily changed
Reusability: Its parts can be used in other projects, so reprogramming is not needed

10. What are the symptoms of the present software crisis? What factors have contributed to the making of the present software crisis?

Software products:
-fail to meet user requirements.
-frequently crash.
-expensive.
-difficult to alter, debug, and enhance.
-often delivered late.
-use resources non-optimally.

Factors affecting such crises are:
- Larger problems,
- Lack of adequate training in software engineering,
- Increasing skill shortage,
- Low productivity improvements

11. Distinguish between generic and customized software products. Which one would generate more revenue for a company? Give reasons.

See question 8
Customized software products require more development effort but have low number of copies in use.

12. What do you understand by the principles of abstraction and decomposition? Why these principles are considered important in software engineering?

Abstraction: Simplify a problem by omitting unnecessary details. Focuses attention on one aspect of the problem.
Decomposition: Decompose a problem into many small independent parts. Each small part is easy to understand and can be easily solved.

These are important as they help solve complex programming problems by breaking them down into simpler ones.

13. What are the types of software? State their differences.

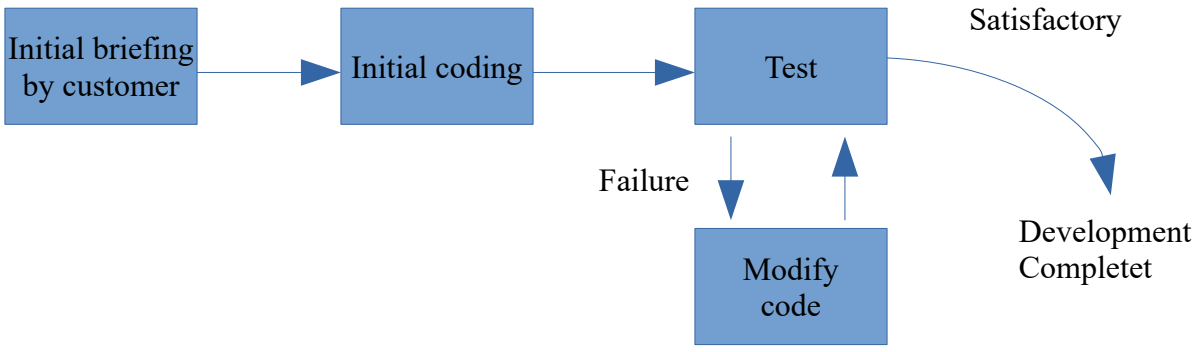
Custom
- For a specific customer

Generic
- Sold on open market
- Often called:
  - COTS (Commercial Off The Shelf)
  - Shrink-wrapped

Embedded
- Built into hardware
- Hard to change

14. What is exploratory style of software development? Explain with diagram.

Also called the "build and fix" style, normally a 'dirty' program is quickly developed. The different imperfections that are subsequently noticed are fixed.



15. How is software different from other products?

- Intangible in nature: hard to understand development
- Easy to reproduce, but cost is in its development(in other products manufacturing is costly)
- Hard to automate, so it is labor-intensive

16. What do you mean by software quality?

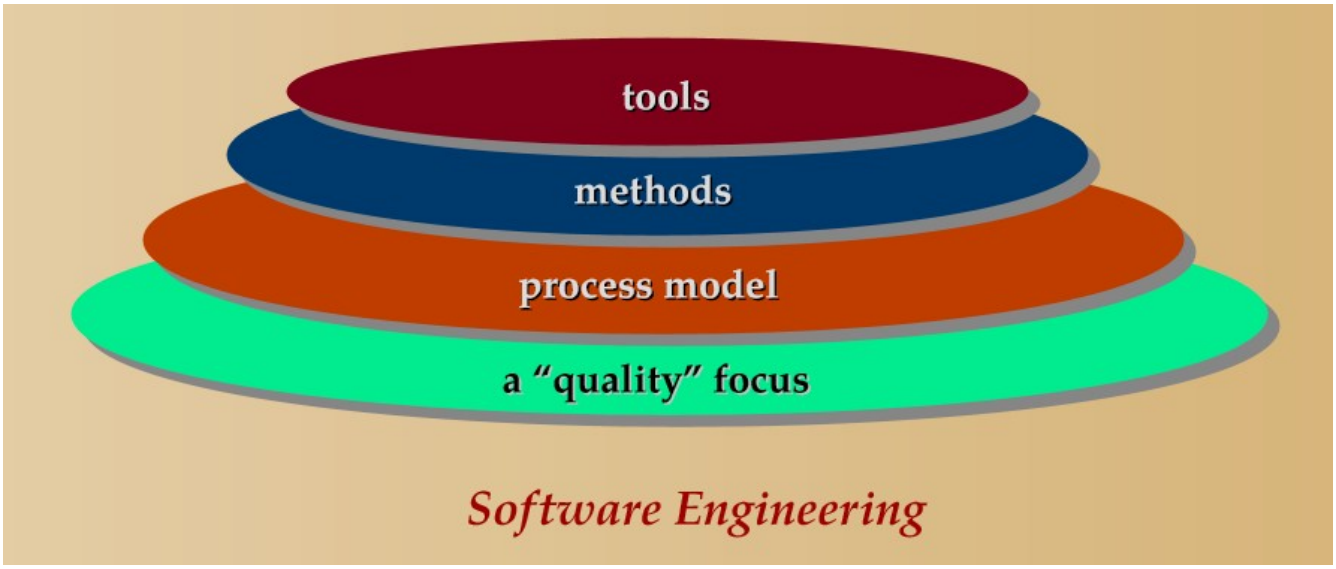
17. Why should we study software engineering?

- To acquire skills to develop large programs.
  - Exponential growth in complexity and difficulty level with size.
  - The ad hoc approach breaks down when size of software increases.
- Ability to solve complex problems
  - How to break down large projects into smaller, more manageable parts
- Learn techniques of specification, design, interface development, testing, project management, etc.

18. What is generic software? Give examples.

Generic software is sold in an open market. It is often called Commercial-off-the-shelves (COTS).
Examples: Chrome, Office, etc

19. Explain Layer technology in software engineering



S/W engineering consists of a process, a set of methods for managing and developing the software, and a collection of tools.
The bedrock that supports S/W engineering is a “quality focus”. It helps define the degree of usability and maintainability, and is a very important parameter in developing S/W.

20. What are Unit testing, Integration testing and System testing?

Unit Testing: Tests for each individual module in the system.
Integration Testing: Tests that check how well the modules work together. i.e. the interfaces between the modules are tested.
System Testing: The entire environment is tested at the client side.

21. What is phase containment of errors?

The principle of detecting errors as close to its point of introduction as possible is called phase containment of errors.

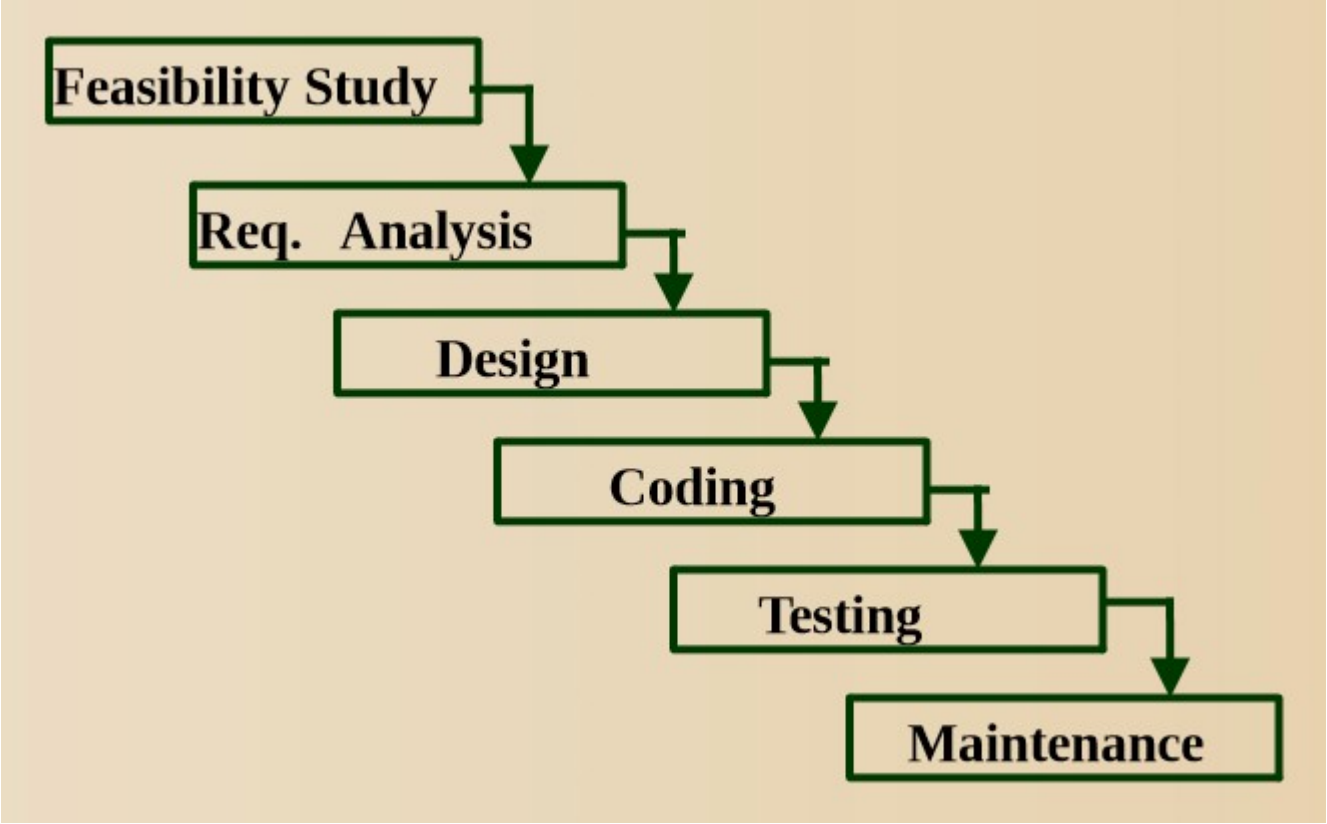
22. What is the difference between function oriented and object oriented design?

Object oriented design models real-world objects and their interactions.

23. What is the need of design phase in SDLC? Explain object oriented design principle.

SDLC

1. Which phases are called development phases in SDLC? Which development phase requires maximum time? Justify your answer.



Phases between Feasibility Study and Testing are called development phases.
Testing phase requires maximum time among development phases as the software has to be correct according to the original requirement specifications.

2. Explain the importance of feasibility study phase of SDLC. What activity is performed during this phase?

The main goal of feasibility study is to work out whether the developing the product is:

- financially worthwhile
- technically feasible

Activities performed:

- Work out an overall understanding of the solution
- Formulate different solution strategies
- Examine alternate solution strategies in terms of resources required, cost of development, and development time.
- Perform a cost/benefit analysis to select the best strategy.

3. Define unit testing, integration testing and system testing. Explain importance of each of them.

Unit Testing: Tests if each individual module in the system works correctly.
Integration Testing: Tests if the partially integrated system is working correctly.
System Testing: The entire environment is tested at the client side.

System testing ensures that the developed system functions according to its requirements as specified in the SRS document.

4. Why the maintenance phase consumes most of the time in SDLC? Describe different maintenance schemes.

Corrective maintenance:

- Correct errors which were not discovered during the product development phases.

Perfective maintenance:

- Improve implementation of the system
- Enhance functionalities of the system.

Adaptive maintenance:

- Port software to a new environment, e.g. to a new computer or to a new operating system.

5. Explain the limitations of classical water fall SDLC model.

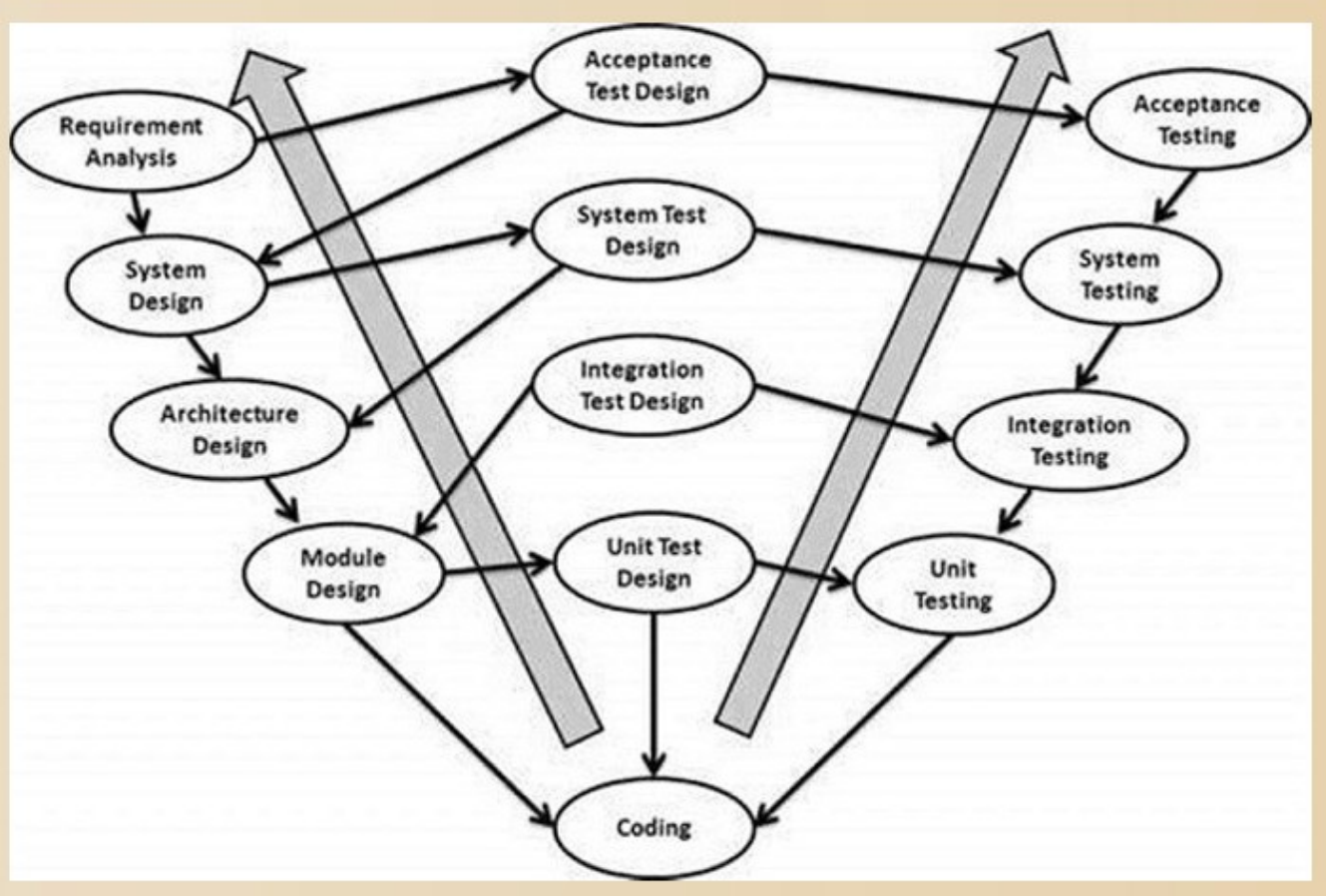
Classical waterfall model is idealistic:

- assumes that no defect is introduced during any development activity.
- in practice, defects do get introduced in almost every phase of the life cycle.
- Defects usually get detected much later in the life cycle
- Once a defect is detected
  - we need to go back to the phase where it was introduced
  - redo some of the work done during that and all subsequent phases.

6. Explain why the waterfall model of SDLC is not suitable for a Web Information System.

primarily due to the pressure of implementing a Web Information System project quickly; the continual evolution of the project requirements; the need for experienced, flexible team members drawn from multiple disciplines; and the inability to make assumptions regarding the users' knowledge level.

7. How is the V-Model of SDLC is different from classical water fall model? Describe the strength and weaknesses of V-Model.



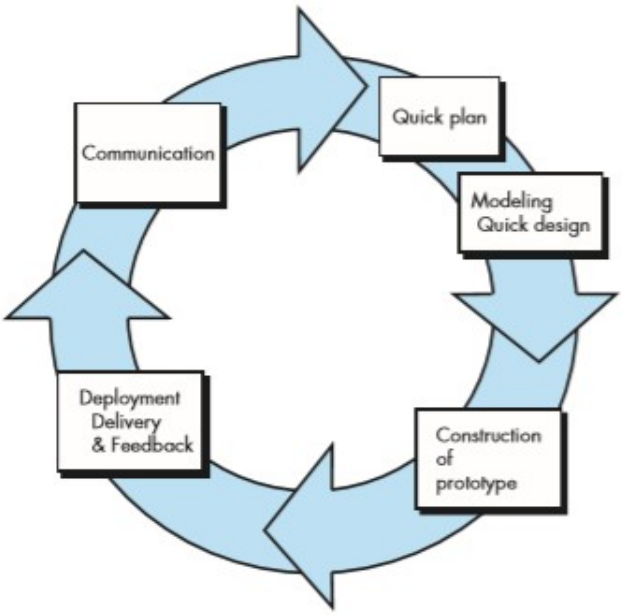
The V-model provides a way of visualizing how verification and validation actions are applied to earlier engineering work.

Strengths:
- As a software team moves down the left side of the V, basic problem requirements are refined into progressively more detailed and technical representations of the problem and its solution.
- Once code has been generated, the team moves up the right side of the V, essentially performing a series of tests (quality assurance actions) that validate each of the models created as the team moves down the left side.

Weaknesses:
- Real projects rarely follow the sequential flow that the model proposes.
- It is often difficult for the customer to state all requirements explicitly, which is required by the water fall model.
- The linear nature of the classical waterfall model leads to “blocking states” in which some team members must wait for other members to finish some dependent tasks.

8. Explain phases for a prototype model. In which case this model is suitable?

FIGURE 4.4
The prototyping paradigm



Communication: You meet with other stakeholders to define the overall objectives for the software, identify whatever requirements are known, and outline areas where further definition is mandatory.
Quick Plan: A prototyping iteration is planned quickly.
Modeling Quick design: A quick design focuses on a representation of those aspects of the software that will be visible to end users.
Construction of Prototype: The quick design leads to a prototype.
Deployment Delivery & Feedback: The prototype is deployed and evaluated by stakeholders, who provide feedback that is used to further refine requirements.

Ideally, the prototype serves as a mechanism for identifying software requirements.

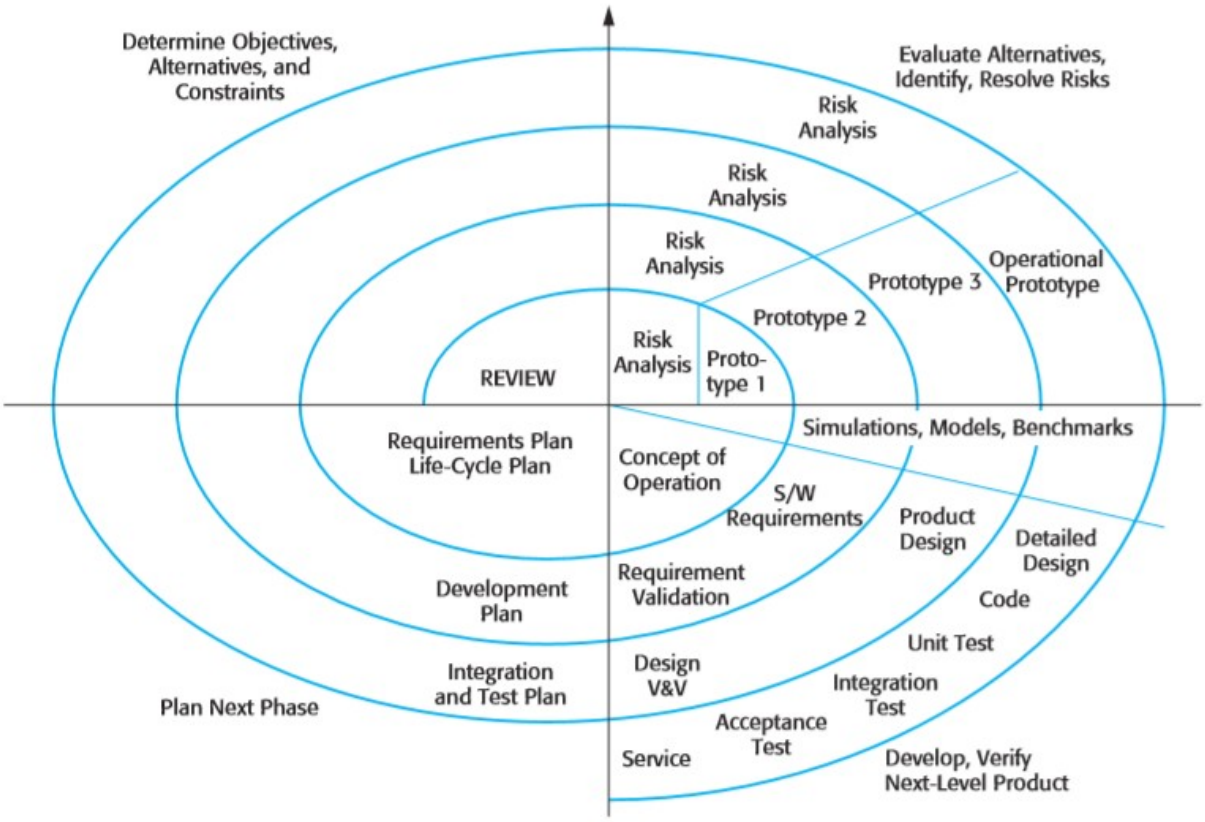
9. Compare Prototype model with incremental model of SDLC.

10. Define the most appropriate and least appropriates cases to use evolutionary model of SDLC.

Appropriate:
Often, a customer defines a set of general objectives for software, but does not identify detailed requirements for functions and features. In other cases, the developer may be unsure of the efficiency of an algorithm, the adaptability of an operating system, or the form that human-machine interaction should take. In these, and many other situations, an evolutionary paradigm may offer the best approach.

Inappropriate:
When time to market is the most important management requirement. i.e. when a market window is missed, the product may be meaningless.

11. Explain four quadrants of Spiral model SDLC.



1. Objective setting: Specific objectives for that phase of the project are defined. Constraints on the process and the product are identified and a detailed management plan is drawn up. Project risks are identified. Alternative strategies, depending on these risks, may be planned.

2. Risk assessment and reduction: For each of the identified project risks, a detailed analysis is carried out. Steps are taken to reduce the risk. For example, if there is a risk that the requirements are inappropriate, a prototype system may be developed.
3. Development and validation: After risk evaluation, a development model for the system is chosen. For example, throwaway prototyping may be the best development approach if user interface risks are dominant. If safety risks are the main consideration, development based on formal transformations may be the most appropriate process, and so on. If the main identified risk is sub-system integration, the waterfall model may be the best development model to use.
4. Planning: The project is reviewed and a decision made whether to continue with a further loop of the spiral. If it is decided to continue, plans are drawn up for the next phase of the project.

12. Compare RAD Model Vs Agile model.

// From google images

RAD	Agile
Based on designing prototypes and then reengineering them into production quality code	Does not allow prototypes
Developers focus by first creating a quick and dirty solution and then improving the code	Break the solution into features
Managed by a project manager	Team members are self managing
Work as individuals, resulting in poor code	Focus on team communication and designing as a group

13. Compare Exploratory programming vs Agile Model.

14. Compare waterfall model vs Agile Model.

15. What is the similarity between incremental system integration and spiral model of design?

//Each loop in the spiral represents an iteration of incremental system integration.

16. For which software development project is waterfall model not recommended?

- Large projects where the requirements are not well understood or are changing for any reasons such as external changes, changing expectations, budget changes or rapidly changing technology.
- Web Information Systems (WIS) primarily due to the pressure of implementing a WIS project quickly; the continual evolution of the project requirements; the need for experienced, flexible team members drawn from multiple disciplines; and the inability to make assumptions regarding the users' knowledge level.
- Real-time systems.
- Event-driven systems.

17. List application problems related to waterfall development

See Q5

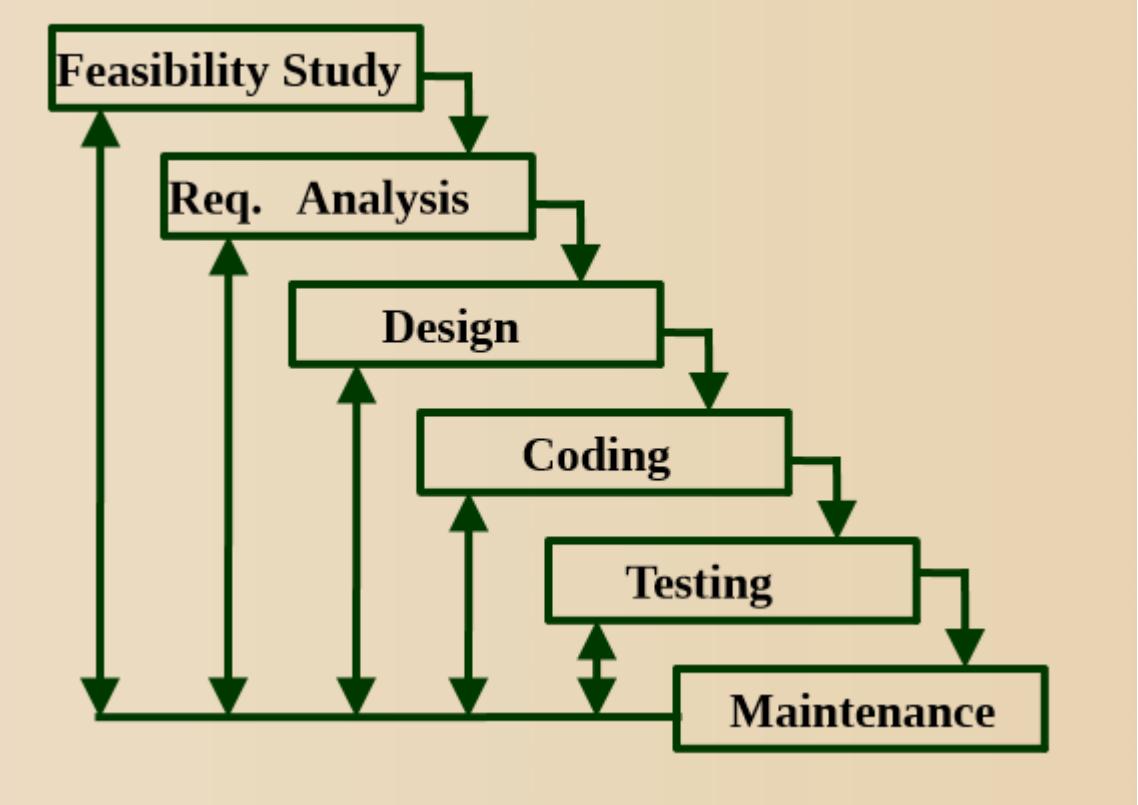
18. What are the main activities of software design phase?

It transforms requirements specifications into a form suitable for implementation in some programming language.

19. Explain the role of prototypes in the evolutionary development.

Prototypes are used to demonstrate concepts, try out design options, and find out more about the problem and its possible solutions. Once developed, the prototype is submitted to the customer for feedback. This is repeated until the user approves the prototype.
The actual product is developed using classical waterfall approach.

20. Explain the process of iterative software development. What are the disadvantages of iterative developments?



Disadvantages:
Errors should be detected in the same phase in which they are introduced.
If a problem is detected in design phase, its easier to fix it there than if it were to be identified at the end of testing.

21. According to you which SDLC model is the best and why?

22. What is feasibility study? Name the different types of feasibility study.

23. Give an example of software product development for which iterative waterfall model is not suitable.

Web Information Systems

24. Which model may be suitable for "An object oriented software development" and why?

Evolutionary model

25. What are the advantages of first constructing a working prototype then the development of actual software product?

- Illustrate input data formats, messages, reports, or interactive dialogs to the customer.
- Examine technical issues such as major design decisions depending on response time of a hardware controller, or the efficiency of an algorithm, etc.
- It is impossible to “get it right” the first time, so plan to throw out the prototype code.

26. What do you understand by the term lifecycle model of software development? List out different SDLC models.

1. Waterfall model
2. Iterative Waterfall model
3. Spiral model
4. RAD model

27. Which are the major phases of the waterfall model of software development? Which phase consumes the maximum effort for developing a typical software product?

Duplicate of Q1

28. What is prototype? What are the major advantages of first constructing a working prototype before developing the actual product?

Duplicate of Q25

29. What is meant by 99 percent complete syndrome? How can it be overcome?

30. What problems of iterative waterfall model have been overcome by V-model? Explain with diagrams.

- As a software team moves down the left side of the V, basic problem requirements are refined into progressively more detailed and technical representations of the problem and its solution.
- Once code has been generated, the team moves up the right side of the V, essentially performing a series of tests (quality assurance actions) that validate each of the models created as the team moves down the left side.

31. Explain CASE tools?

32. Differentiate validation and verification?

33. Do the Comparison of Different Life Cycle Models?

34. Which is the most important feature of Spiral Model?

35. What is Prototype Model in SDLC? Where is this model used?

Requirements

1. What is requirements engineering? Explain the activities performed in requirement analysis and specification phase of SDLC.

The requirements for a system are the descriptions of what the system should do— the services that it provides and the constraints on its operation. These requirements reflect the needs of customers for a system that serves a certain purpose such as controlling a device, placing an order, or finding information. The process of finding out, analyzing, documenting and checking these services and constraints is called requirements engineering.

2. Explain the need and use of a SRS document?

The SRS document is an official statement of what the system developers should implement. It should include both the user requirements for a system and a detailed specification of the system requirements.
It acts as a contract between the clients and the developers.

3. Who are the stake holders of SRS document?

- System Customers – Who specify the requirements and ensure they meet their needs.
- Managers – Use it to plan the cost of the project and the process.
- System Engineers – Use it to understand what is to be developed.
- System Test Engineers – Use it to develop validation tests for the system.
- System Maintenance Engineers – Use it to understand the system and the relationships between the parts.

4. Differentiate between functional and non-functional requirements.

Functional

- Describes what the system has to do
- What are the expectations from the software?
- What the software should not do

Non-functional

- Mostly quality requirements
- Highlights how well the software performs its function
  - for users: high performance, reliability, usability
  - for developers: maintainability, testability, and codeability

5. Write down 3 functional requirements with I/O for a Bank ATM software?

- A user shall be able to search the appointments lists for all clinics.
- The system shall generate each day, for each clinic, a list of patients who are expected to attend appointments that day.
- Each staff member using the system shall be uniquely identified by his or her 8-digit employee number

6. Describe the importance of non-functional requirements.

Non-functional requirements place restrictions on the product being developed, the development process, and specify external constraints that the product must meet.
If these are not met, the system may be useless.

7. Explain the non-functional requirements relevant to a critical system.

8. Describe the measurable metrics for non-functional requirements?

9. Define and explain software efficiency.

10. Write down some trigger questions for performance characteristics, quality issues and resource management issues.

- Performance
  - Are there any speed, throughput, or response time constraints on the system?
  - Are there size or capacity constraints on the data to be processed by the system?
- Quality
  - What are the requirements for reliability?
  - What is the maximum time for restarting the system after a failure?
  - What is the acceptable system downtime per 24-hour period?
- Resource Management
  - How often will the system be backed up?
  - Who will be responsible for the back up?
  - Who is responsible for system installation?
  - Who will be responsible for system maintenance?

11. Write a note on IEEE standard for SRS document.

// Figure 4.7, Sommerville book

Preface	This should define the expected readership of the document and describe its version history, including a rationale for the creation of a new version and a summary of the changes made in each version.
Introduction	This should describe the need for the system. It should briefly describe the system's functions and explain how it will work with other systems. It should also describe how the system fits into the overall business or strategic objectives of the organization commissioning the software.
Glossary	This should define the technical terms used in the document. You should not make assumptions about the experience or expertise of the reader.
User requirements definition	Here, you describe the services provided for the user. The non-functional system requirements should also be described in this section. This description may use natural language, diagrams, or other notations that are understandable to customers. Product and process standards that must be followed should be specified.
System Architecture	This chapter should present a high-level overview of the anticipated system architecture, showing the distribution of functions across system modules. Architectural components that are reused should be highlighted.
System requirements specification	This should describe the functional and non-functional requirements in more detail. If necessary, further detail may also be added to the non-functional requirements. Interfaces to other systems may be defined.
System Models	This might include graphical system models showing the relationships between the system components, the system, and its environment. Examples of possible models are object models, data-flow models, or semantic data models.
System Evolution	This should describe the fundamental assumptions on which the system is based, and any anticipated changes due to hardware evolution, changing user needs, and so on. This section is useful for system designers as it may help them avoid design decisions that would constrain likely future changes to the system
Appendices	These should provide detailed, specific information that is related to the application being developed; for example, hardware and database descriptions. Hardware requirements define the minimal and optimal configurations for the system. Database requirements define the logical organization of the data used by the system and the relationships between data.
Index	Several indexes to the document may be included. As well as a normal alphabetic index, there may be an index of diagrams, an index of functions, and so on

12. What should be included in the Functional Requirements Document?

13. "Consider a system where, a heat sensor detects an intrusion and alerts the security company." What kind of a requirement the system is providing? What kind of a requirement the system is providing (Functional or Non-Functional)?

14. A set of requirements are given below. Identify the type of requirement (functional/ Nonfunctional) a. Field 1 accepts numeric data entry. b. Field 2 only accepts dates before the current date. c. Screen 1 can print on-screen data to the printer.

15. Identify three functional requirements for the product, PATIENT MONITORING SYSTEM.

16. What is portability? Identify requirements related to portability for the product, PATIENT MONITORING SYSTEM.

Portability is the degree to which software running on one platform can easily be converted to run on another.

17. Requirements analysis is critical to the success of a development project. Justify.

18. What are User Interface Requirements?

19. How many types of software requirements are there? Specify them?

Functional Requirements

Business Rules
Transaction corrections
Administrative functions
Authentication
Authorization
Audit Tracking
External Interfaces

Non-functional Requirements

Product requirements

- Usability requirements
- Reliability requirement
- Safety requirements
- Efficiency requirements
- Performance requirements
- Capacity requirements

Process requirements

- Delivery requirements
- Implementation requirements
- Standards requirements

External requirements

- Legal constraints
- Economic constraints
- Interoperability requirements

20. How are the abstraction and decomposition principles used in developing a good requirement specification?

21. List five desirable characteristics of a good Software Requirements specification (SRS) document.

Consistent
Correct
Modifiable
Unambiguous
Complete
Understandable
Verifiable
Basis for design and testing
Act as a contract
Traceable

22. What is functional and non-functional requirements of a system? How to identify the functional requirements? Explain with suitable example.

Functional

- Describes what the system has to do
- What are the expectations from the software?
- What the software should not do

Non-functional

- Mostly quality requirements
- Highlights how well the software performs its function
  - for users: high performance, reliability, usability
  - for developers: maintainability, testability, and codeability

23. What is a requirement? What are the types of requirements? Explain.

24. According to you, what should be the functional requirements for the mental health carepatient management system?

25. What should ideally be the software requirement specification (SRS) for developing an online shopping app?

26. How can you gather requirements?

27. What is SRS?

28. How do you find functional and non-functional requirements? Identify at least four nonfunctional requirements of the case study, Library Information System.

29. What is the difference between SRS and FRS? Who will create SRS?

30. What does SRS document contain? Prepare a SRS document for Online Grocery Shop.

Android Programming

1. Explain the android application architecture.

Applications consist of the following components:

- activities, which represent entry points for the user's interaction with the app (Each activity represents a screen of the app)
- services are general-purpose entry points for keeping an app running in the background for any kind of reasons.
- broadcast receivers are components that enable the system to deliver events to the app outside of a regular user flow, allowing the app to respond to system-wide broadcast announcements
- content providers manages a shared set of app data that you can store in the file system, in a SQLite database, on the web, or on any other persistent storage location that apps can access.

2. What are the core building blocks of androids?

- Linux kernel
- Libraries – bionic libc, webkit, sqlite, opengl, media framework, etc
- Runtime – Java core libraries, Dalvik VM (now replaced by Android Runtime)
- Application Frameworks – Activity Manager, Content Providers, Notification Manager, etc
- Applications – Core set of applications shipped with the platform such as email, browser, etc

3. Discuss the major attributes of the TextView.

- android:text – The text to show in the view. It is recommended to use a string resource if the value is not dynamic.
- android:textSize – Font size of the text, in sp.
- android:fontFamily – Font used for the text that is shown.

4. Differentiate between an ImageView and an ImageButton.

ImageView	ImageButton
Displays image resources.	Displays a button with an image without text.
Not interactive.	Has the pressed,selected, focused, unselected states that can convey to the user the different states of the button.

5. Define Intents in android? What are the different types of Intents?

An Intent is a messaging object used to request an action from another app component. Intents are used to launch activities, start services, display a web page, etc.

- Implicit Intent - doesn't specify the component. In such case, intent provides information of available components provided by the system that is to be invoked.
- Explicit Intent - specifies the component. In such case, intent provides the external class to be invoked.

6. What are the different types of app components in android studio? Explain.

See Q1

7. What is the difference between constraint layout and relative layout?

Constraint Layout	Relative Layout

8. What are the ways to assign height and width of widgets in android studio? Explain their difference.

android:height/android:width - Fixed size in pixels(px), device-independent pixels(dp), scaled pixels(sp), inches(in), millimeters(mm)

android:layout_height/android:layout_width – MATCH_PARENT to be as big as the widget's parent, or WRAP_CONTENT to be just big enough for the content of the widget, or a value similar to android:height/android:width.

9. How does a broadcast receiver differ from an implicit intent?

Broadcast Receiver	Implicit Intent
Messages are sent to any app that has defined a receiver.	The message requests an action to be performed, such as handling a web url. Thus it is only delivered to the app accepts the intent.

10. What is an .apk extension in Android? What is its use?

The compiler provided in the Android SDK compiles the application source for running on the Android Runtime (ART). The resulting .apk file bundles the compiled form of the application, with a manifest file describing the components of the application.

From the user's point of view, apk files are used to install applications on an Android system.

11. What is meant by an Event? What are the ways by which an Event Listeners Registration can take place?

Events are a useful way to collect data about a user's interaction with interactive components of Applications like button presses or screen touch etc.
Event Listener Registration can take place:

- Using an Anonymous Inner Class
- Activity class implements the Listener interface
- Using Layout file activity_main.xml to specify event handler directly

12. Write a small code snippet in Java to fetch data from EditText when a button is pressed and display it as Textview.

```
Button b1=(Button)findViewById(R.id.button);
b1.setOnClickListener(new View.OnClickListener() {
        @Override public void onClick(View v) {
                EditText editText = (EditText) findViewById(R.id.editText);
                // Fetch the text
                String text = editText.getText().toString();
        }
});
```

13. How to launch an activity in android?

```
Intent i = new Intent(getApplicationContext(), ActivityTwo.class);
startActivity(i);
```

14. Differentiate implicit and explicit intent with a suitable coding example.

- Implicit Intent - doesn't specify the component. In such case, intent provides information of available components provided by the system that is to be invoked.

```
Intent i=new Intent(Intent.ACTION_VIEW);
i.setData(Uri.parse("http://www.javatpoint.com"));
startActivity(intent);
```

- Explicit Intent - specifies the component. In such case, intent provides the external class to be invoked.

```
Intent i = new Intent(getApplicationContext(), ActivityTwo.class);
startActivity(i);
```

15. Describe the significance of all folders in an android project.

Myapplication (Android Project Name)

- src: source for the activities as java files
- gen: java source files generated by the android sdk
- bin: apk files
- libs: external libraries and jar files
- assets: external fonts, images, videos, etc
- res: resources used by the application
  - drawable: images/icons
  - layout: screen layouts in xml format
  - values: string values
  - menus: application menus
- manifests:
  - androidmanifest.xml: xml file describing the application's components

16. State the importance of Manifest.xml file in android.

- The manifest file describes essential information about the app
- Declares:
  - the app's package name
  - components of the app - activities, services, broadcast receivers, and content providers along with the name of their java classes
  - The permissions that the app needs in order to access protected parts of the system or other apps. It also declares any permissions that other apps must have if they want to access content from this app.
  - The hardware and software features the app requires, which affects which devices can install the app

17. How to open the web browser in android? Write the Corresponding Java code.

To open the user's preferred web browser to view a site, an implicit intent is used:

```
Intent i=new Intent(Intent.ACTION_VIEW);
i.setData(Uri.parse("http://www.javatpoint.com"));
startActivity(intent);
```

18. What are the main components of android architecture? Explain briefly each component.

See Q1

19. What is the importance of XML based Layout?

- Declaring your UI in XML allows you to separate the presentation of your app from the code that controls its behavior.
- Using XML files also makes it easy to provide different layouts for different screen sizes and orientations.

20. What is an Activity? Which method is implemented by all subclasses of an Activity?

The Activity class is a crucial component of an Android app, and the way activities are launched and put together is a fundamental part of the platform's application model. Unlike programming paradigms in which apps are launched with a main() method, the Android system initiates code in an Activity instance by invoking specific callback methods that correspond to specific stages of its lifecycle.

All subclasses of an Activity have to implement the onCreate() function, which fires when the system creates your activity.

21. What is android? What is the latest version of android?

Android is a mobile operating system based on the linux and kernel for mobile devices such as smartphones and tablet computers. It is developed by the Open Handset Alliance led by Google.
The latest version of android is Android 10.

22. What is the difference between File, Class, and Activity in android?

File - It is a block of arbitrary information, or resource for storing information. It can be of any type.

Class - Its a compiled form of .java files. Android SDK uses the .class files to produce an apk

Activity - An activity represents a screen that the user interacts with. It is implemented as a java class.

23. What is a Toast? Write its syntax.

A toast provides simple feedback about an operation in a small popup. It only fills the amount of space required for the message and the current activity remains visible and interactive. Toasts automatically disappear after a timeout.

```
Context context = getApplicationContext();
CharSequence text = "Hello toast!";
Toast toast = Toast.makeText(context, text, Toast.LENGTH_SHORT);
toast.show();
```

24. Can Android applications only be programmed in Java?

Google now offers, and recommends using Kotlin for development of Android applications.
// There are also alternative frameworks for mobile applications, such as React Native,
// Xamarin, Qt, and Flutter.

25. Explain briefly the different Layouts in Android.

- LinearLayout – All elements are shown in a linear manner (horizontal or vertical)
- RelativeLayout – Widgets are placed according to their relationship to the others in the container and their parent container.
- TableLayout – Arranges widgets into rows and columns.
- FrameLayout – Child elements are placed like a stack, with the top of the stack being at the front of the layout
- AbsoluteLayout – Child elements are placed according to specific coordinates. Not recommended as it is inflexible for the large number of devices and form factors that Android supports.
- ConstraintLayout – Places widgets in a manner similar to RelativeLayout but is more flexible.

26. What Is the Google Android SDK?

It is a set of development tools used to develop apps for Android. It includes

- Required set of libraries
- Debugger
- Emulator
- Documentation for the Android APIs
- Samples on using the APIs

27. What is the use of an activityCreator?

activityCreator is a batch file and shell script used to create Android projects. It has long since been replaced by the android create project (Create > New Project in Android Studio).

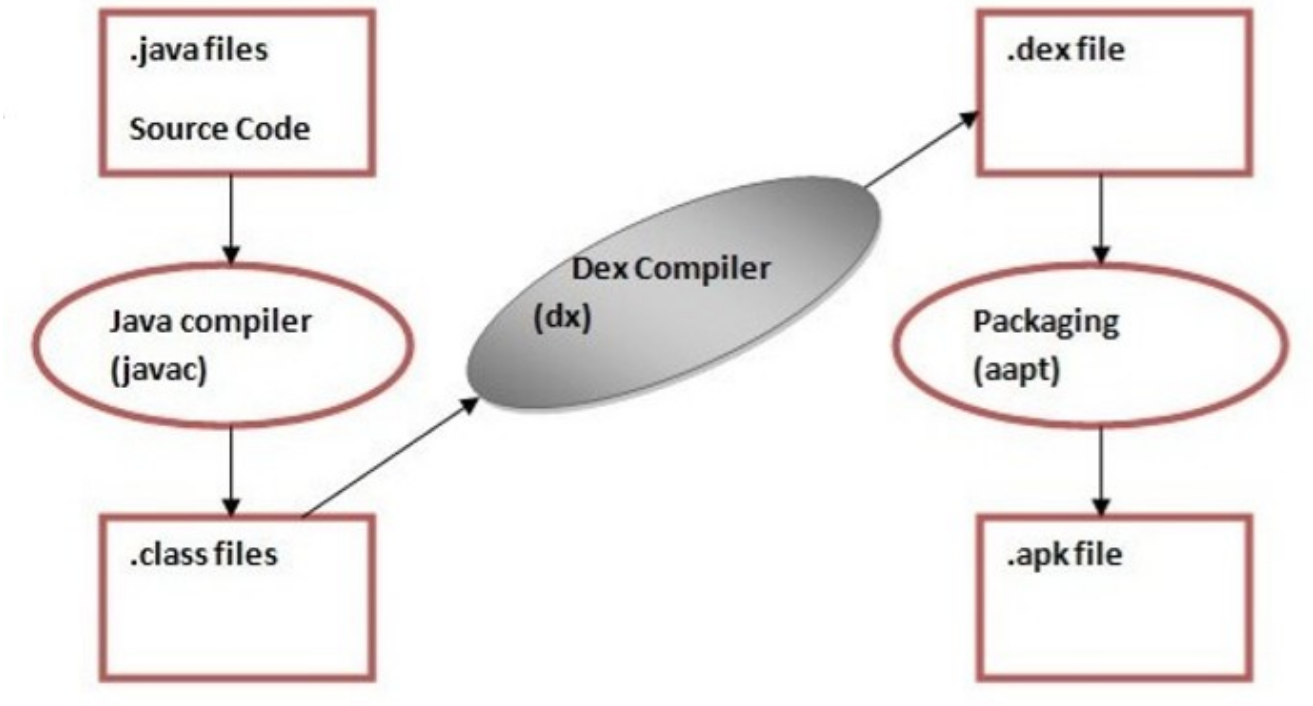
28. List out difference types of layout?

See Q25

29. Explain the Application framework of Android.

- Simplifies the reuse of components
  - Applications can publish their capabilities and any other application may then make use of those capabilities
- Applications is a set of services and systems, including Views system, content providers, resources manager and so on

30. Explain stages used to convert a java program to an apk?



31. Justify that ART is better than DVM.

Android Runtime (ART)	Dalvik VM (DVM)
Applications are compiled Ahead-of-Time (AoT) into native instructions	Applications are compiled Just-in-Time (JiT) when executing on the device
Improves battery efficiency as there is no need to interpret the JiT instructions at runtime	Worse battery efficiency as the JiT instructions must be interpreted with the DVM
Faster execution of applications as they are compiled into native executables	Slower execution of applications as the DVM has to execute the JiT instructions
Improved Garbage collection techniques lead to more efficient memory usage	

32. Explain the functions of MainActivity.java and activity_main.xml, and AndroidManifest.xml. Mention the location of these files.

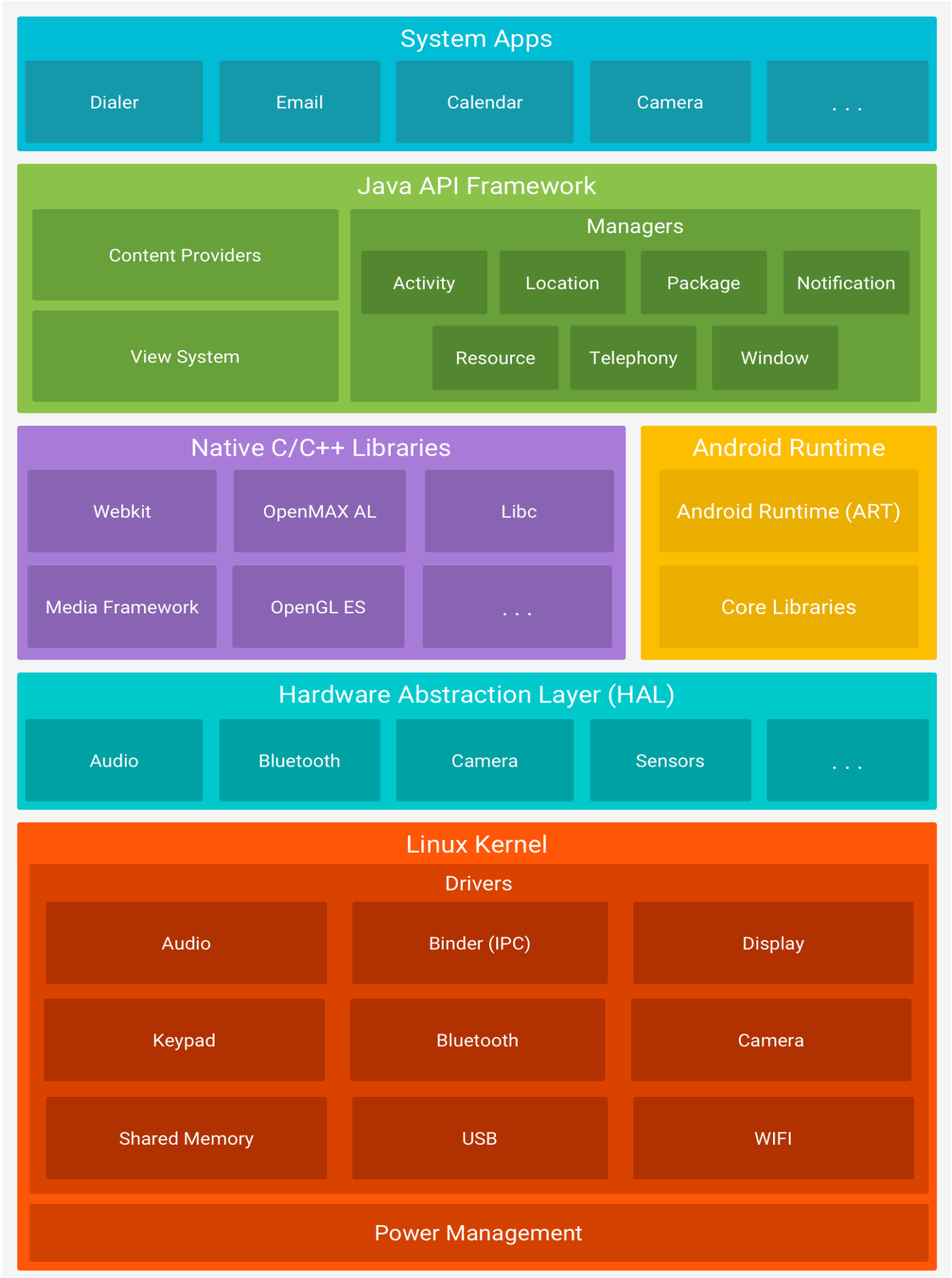
activity_main.xml – Describes the layout of the main activity. Present in src/main/res/layout
MainActivity.java – The source code that is responsible for loading the main activity ui. Present in src/main/java/
AndroidManifest.xml – Describes the application – its components, permissions, etc. Present in src/main/

33. What are the components of Android runtime?

Android runtime consists of:

- Core libraries – java libraries providing main features of the java language.
- Android Runtime(ART)/ Dalvik VM – Virtual machine to execute applications in Dex format

34. Draw the different layers of Android Architecture.


System Apps
Dialer
Email
Calendar
Camera
. . .
Java API Framework
Content Providers
View System
Managers
Activity
Location
Package
Notification
Resource
Telephony
Window
Native C/C++ Libraries
Webkit
OpenMAX AL
Libc
Media Framework
OpenGL ES
. . .
Android Runtime
Android Runtime (ART)
Core Libraries
Hardware Abstraction Layer (HAL)
Audio
Bluetooth
Camera
Sensors
. . .
Linux Kernel
Drivers
Audio
Binder (IPC)
Display
Keypad
Bluetooth
Camera
Shared Memory
USB
WIFI
Power Management

35. Explain the hierarchy used for Android UI.

Activity's UI is defined by a hierarchy of View and ViewGroup nodes.

Views are widgets/ui components such as buttons, text views, etc

ViewGroups are containers that have a layout controlling how views are arranged in it.