### **1. What are the various categories of software?**

Software products are mainly categorized into:

* ****System software:****Softwares like operating systems, compilers, drivers, etc. fall into this category.
* ****Networking and web development software:****Computer networking software offers the necessary functionality for computers to communicate with one another and with data storage facilities.
* ****Embedded Software:****Software used in instrumentation and control applications such as washing machines, satellites, microwaves, TVs, etc.
* ****Artificial Intelligence Software:****Expert systems, decision support systems, pattern recognition software, artificial neural networks, and other types of software are included in this category.
* ****Scientific software:****These support a scientific or engineering user's requirements for performing enterprise-specific tasks. Examples include MATLAB, AUTOCAD, etc.

### **2. What are the characteristics of software?**

There are six major **[characteristics of software](https://www.interviewbit.com/blog/what-are-the-characteristics-of-software/" \t "https://www.interviewbit.com/software-engineering-interview-questions/_blank)**:

* ****Functionality****: The things that software is intended to do are called functionality. For example, a calculator's functionality is to perform mathematical operations.
* ****Efficiency:**** It is the ability of the software to use the provided resources in the best way possible. Increasing the efficiency of software increases resource utilization and reduces cost.
* ****Reliability:****Reliability is the probability of failure-free operational software in an environment. It is an important characteristic of software.
* ****Usability:**** It refers to the user’s experience while using the software. Usability determines the satisfaction of the user.
* ****Maintainability:****The ease with which you can repair, improve, and comprehend software code is referred to as maintainability. After the customer receives the product, a phase in the software development cycle called software maintenance begins.
* ****Portability:**** It refers to the ease with which the software product can be transferred from one environment to another.

Apart from the above-mentioned characteristics, the software also has the following characteristics:

* Software is engineered, it is not developed or manufactured like hardware. Development is an aspect of the hardware manufacturing process. Manufacturing does not exist in the case of software.
* The software doesn't wear out.
* The software is custom-built.

### **3. What is a framework?**

A framework is a well-known method of developing and deploying software. It is a set of tools that allows developing software by providing information on how to make it on an abstract level, rather than giving exact details. The Software Process Framework is the basis of the entire software development process. The umbrella activities are also included in the software process structure.

### **4. What is the main difference between a computer program and computer software?**

The key difference between software is a collection of several programs used to complete tasks, whereas a program is a set of instructions expressed in a programming language. A program can be software, but software the vice versa is not true.

### **5. Describe the Software Development Process in Brief.**

The **[Software Development Life Cycle (SDLC)](https://www.interviewbit.com/sdlc-interview-questions/" \t "https://www.interviewbit.com/software-engineering-interview-questions/_blank)** is a number of fundamental phases that teams must follow in order to produce and deliver high-quality software. Software typically goes through the following phases:

* ****Requirements Gathering:****The team identifies, collects, and defines core issues, requirements, requests, and customer expectations related to the software application or service during this stage of the project. Generating software specifications, creating a thorough strategy, documentation, issue tracking, and project or product planning, including allocating the resources, are some tasks done during this phase.
* ****Software Design:**** The team makes software design decisions regarding the architecture and make of the software solution throughout this design phase of development.
* ****Software Development:**** Teams develop software solutions based on the design decisions made during earlier stages of the project
* ****Testing and Integration:**** Software is packaged and tested to ensure quality. Quality assurance, often known as testing, ensures that the solutions deployed fulfil the specified quality and performance criteria.
* ****Deployment:**** The software is installed in a production setting. The gathered, designed, built, and tested work is shared with the software service's customers and users.
* ****Operation and Maintenance:**** The software is installed in a production setting. The work is shared with the software service's customers and users.

### **6. What are different SDLC models available?**

* **[Waterfall model](https://www.interviewbit.com/blog/waterfall-model-in-software-engineering/" \t "https://www.interviewbit.com/software-engineering-interview-questions/_blank)**
* **[Spiral model](https://www.interviewbit.com/blog/spiral-model/" \t "https://www.interviewbit.com/software-engineering-interview-questions/_blank)**
* **[Incremental model](https://www.interviewbit.com/blog/incremental-model/" \t "https://www.interviewbit.com/software-engineering-interview-questions/_blank)**
* **[Agile Model](https://www.interviewbit.com/blog/agile-model/" \t "https://www.interviewbit.com/software-engineering-interview-questions/_blank)**
* **[Big bang model](https://www.interviewbit.com/blog/big-bang-model/" \t "https://www.interviewbit.com/software-engineering-interview-questions/_blank)**
* **[Iterative model](https://www.interviewbit.com/blog/iterative-model/" \t "https://www.interviewbit.com/software-engineering-interview-questions/_blank)**

### **7. Which SDLC model is the best?**

According to the annual State of Agile report, Agile is the best SDLC methodology and also one of the most widely used SDLC in the IT industry. The reason is that it is a hybrid of incremental and iterative approaches and works well in a flexible environment. That being said, select the model that suits your requirements.

### **8. What is Debugging?**

Debugging is the process of finding a software bug, in the context of software engineering. To put it another way, it refers to the process of finding, evaluating, and correcting problems. Debugging is essential once the software fails to run properly.

### **9. What does a software project manager do?**

A software product manager leads and manages the software product management department. They are in charge of the software product's specialization, goals, structure, and expectations. They also head the planning, backlog grooming, stakeholder management, and providing roadmap necessary to create the best software.

### **10. What is the waterfall method and what are its use cases?**

The waterfall is the easiest and most straightforward SDLC approach in software development. In this approach, the development process is linear, and each step is finished one by one. As the name implies, development progresses downwards, much like a waterfall. The software has to cover the following phases in a waterfall model:

* Requirements
* Design
* Implementation
* Testing and integration
* Deployment
* Maintenance

****Use cases:****

* When requirements are well-defined and unchangeable.
* There are no ambiguous requirements or conditions.
* When the technology is well understood
* The project is brief, and the cast is small.
* The risk is negligible.

### **11. What is Software scope?**

The scope of a software project is a well-defined boundary that incorporates all the activities involved in developing and delivering a software product.  The scope defines what the product will and will not do, as well as what the final product will and will not contain. All capabilities and objects to be delivered as part of the software are explicitly defined in the software scope.

### **12. What are the merits of the incremental model?**

* It can deliver iteration faster, in the first iteration itself.
* Development takes place in parallel to each other.
* We can reduce the first delivery cost by using this method.
* The user or client can provide feedback at each level and unexpected changes in the requirement can be avoided.
* Risks can be identified and managed on a module-by-module basis.

### **13. What is Software prototyping and POC?**

A software prototype is a working model with limited functionality. The prototype may or may not contain the exact logic used in the final software program, and therefore is an additional work that should be considered in the calculation. Users can review developer proposals and try them out before they are implemented through prototyping. It also helps in comprehending user-specific details that may have been missed by the developer during product development.

POC (Proof of Concept) is a method used by organizations to validate an idea or concept's practicality. The stage exists prior to the start of the software development process. On the basis of technical capability and business model, a mini project is built to see if a concept can be executed.

### **14. What are the drawbacks of the spiral model?**

The spiral model is a hybrid of the iterative development process and the waterfall model, with a focus on risk analysis. In the SDLC Spiral model, the development process begins with a limited set of requirements and progresses through each development phase.  Until the application is ready for production, the software engineering team adds functionality for the increased requirement in ever-increasing spirals.

****Drawbacks of the spiral model are****:

* It's significantly more complicated than other SDLC models. The procedure is intricate.
* Due to its high cost, it is not recommended for small projects.
* Risk Analysis is overly reliant, and it necessitates a high level of skill.
* Time estimation is challenging
* The spiral could continue endlessly.

### **15. What is baseline in Software Development?**

A baseline is a software development milestone and reference point marked by the completion or delivery of one or more software deliverables. The main objective of the baseline is to decrease and regulate vulnerability, or project weaknesses that can easily damage the project and lead to uncontrollable changes.

### **16. What is SRS?**

SRS is a formal report that serves as a representation of software that allows customers to assess whether it meets their needs. It is a list of requirements for a certain software product, program, or set of apps that execute specific tasks in a specific environment. It also includes user needs for a system, as well as precise system requirements specifications. Depending on who is writing it, it fulfils a variety of purposes.

### **17. What are CASE tools?**

CASE tools are a collection of software application programs that automate SDLC tasks. Analysis tools, Design tools, Project management tools, Database Management tools, and Documentation tools are a few of the CASE tools available to simplify various stages of the Software Development Life Cycle.

### **18. What are Verification and Validation?**

* ****Verification:****The process of ensuring that software accomplishes its objectives without defects is known as verification. It's the procedure for determining whether the product being developed is correct or not. It determines whether the resulting product meets our specifications. It is mainly focused on functionality.
* ****Validation:****Validation is the process of determining whether a software product meets the required standards, or in other words, whether it meets the product's quality criteria. It is the process of verifying product validation or ensuring that the product we are building is correct. Validation focuses on the quality of the software.

### **19. What do you mean by Software Re-engineering?**

The process of updating software is known as software reengineering. This procedure entails adding new features and functionalities to the software in order to make it better and more efficient.

## **Software Engineering Interview Questions for Experienced**

### **20. What is the feasibility study?**

As the name implies, a feasibility study is a measurement of a software product in terms of how useful product development will be for the business from a practical standpoint. Feasibility studies are conducted for a variety of reasons, including determining whether a software product is appropriate in terms of development, implementation, and project value to the business. The feasibility study concentrates on the following areas:

* Economic feasibility
* Technical feasibility
* Operational feasibility
* Legal feasibility
* Schedule feasibility

### **21. Define black box testing and white box testing?**

* ****Black box testing**** is a type of high-level testing in which the primary goal is to evaluate functionalities from a behavioural standpoint. In black-box testing, the tester does not test the code; instead, they utilize the program to see if it works as expected.
* When you have insight into the code or broad information about the architecture of the software in question, you can perform ****white box testing****, also known as ****clear box testing****. It falls under the category of low-level testing and is mostly concerned with integration and unit testing.  White box testing requires programming expertise or at the very least a thorough grasp of the code that implements a particular functionality.

### **22. What is Concurrency?**

In software engineering, concurrency refers to a set of techniques and mechanisms that allow the software to do many tasks at the same time. Concurrency can be achieved by using languages like C++ or Java because these languages support the concept of thread. New hardware and software features are required to achieve concurrency.

### **23. What are Software Metrics?**

A software metric is a quantitative measure of program properties. Software metrics can be used for a range of things, such as analyzing software performance, planning, estimating productivity, and so on. Load testing, stress testing, average failure rate, code complexities, lines of code, etc. are some software metrics. The benefits of software metrics are many, some of them being:

* It reduces cost.
* It increases ROI (return on investment).
* Reduces workload.
* Highlights areas for improvement.

### **24. What is the difference between cohesion and coupling?**

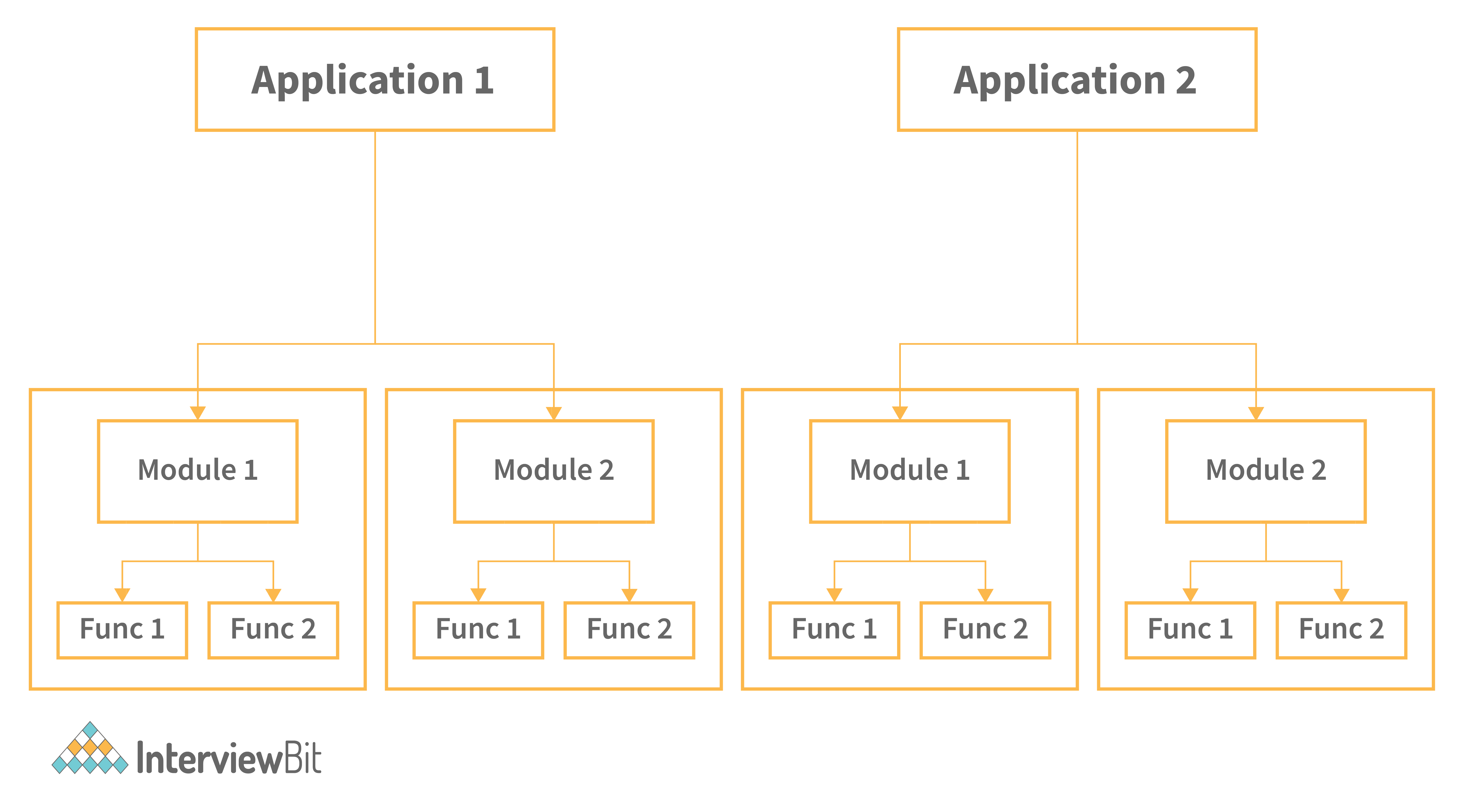
| **Cohesion** | **Coupling** |
| --- | --- |
| Cohesion refers to the relationship within modules. | Coupling refers to the relationship between modules. |
| Increasing cohesion is good for the software. | Coupling should be avoided. |
| Modules focus on a particular thing in cohesion. | Modules are coupled to one another through coupling. |
| ****Example:****A function that checks file permission and then opens it, or a function to decrypt messages. | ****Example:****Two models sharing data with each other. |

### **25. What is Data Flow Diagram?**

A ****Data Flow Diagram**** (DFD) shows the flow of information flows through a system. It shows data inputs, outputs, storage sites, and paths between each destination using symbols such as rectangles, circles, and arrows, as well as short text labels. Data flowcharts can range from simple to in-depth DFDs that go deeper into how data is processed. They can be used to evaluate a current system or to create a new system. A DFD can effortlessly express things that are difficult to describe in words, and it can be used by both technical and non-technical audiences.

### **26. Explain the concept of modularization.**

Modularization is ‌breaking down a program's functionality into separate, independent modules, each of which includes just the information needed to carry out one part of the intended capability. In simple terms, it is the practice of dividing the program into smaller modules so that we can deal with them separately. We can simply add independent and smaller modules to a program using modularization without being hampered by the complexity of the program's other functionalities. Modularization is based on the notion of designing applications that are easier to develop and maintain, self-contained components. In monolithic design, on the other hand, there's always the risk of a simple change knocking the entire application down. The final step would be to combine these independent modules.



In the above diagram, both the applications have been divided into smaller modules. These modules can then be dealt with separately.

### **27. What is Software Configuration Management?**

When a piece of software is created, there is always room for improvement. To modify or improve an existing solution or to establish a new solution for a problem, changes may be required. Changes to the existing system should be examined before being implemented, recorded before being implemented, documented with details of before and after, and controlled in a way that improves quality and reduces error. This is where System Configuration Management is required.

During the Software Development Life Cycle, Software Configuration Management (SCM) is a technique for systematically managing, organizing, and controlling changes in documents, codes, and other entities. The main goal is to enhance production while making as few mistakes as possible.

### **28. What are functional and non-functional requirements?**

| **Functional Requirements** | **Non-functional Requirements** |
| --- | --- |
| These are the needs that the end-user specifies as essential features that the system should provide. | These are the quality requirements that the system must meet in order to fulfil the project contract. |
| The user specifies the functional requirements. | Technical individuals, such as architects, technical leaders, and software engineers, specify non-functional requirements. |
| Functional Requirements are mandatory. For example, the client might want certain mandatory changes in UI, like dark mode. | Non-functional requirements are not Mandatory. For example, the requirement to enhance readability is non-functional. |

### **29. What is the difference between Quality Assurance and Quality control?**

| **Quality Assurance** | **Quality Control** |
| --- | --- |
| Quality Assurance focuses on assuring that the end product (software) will be of the requested quality. | Quality control focuses on controlling the processes, methods, or techniques used in the development of software so that the quality requested is fulfilled. |
| It is a preventive measure. | It is a corrective measure. |
| It applies to the full software development life cycle. | It is applied in the testing phase. |

### **1) What are the important categories of software?**

* System software
* Application software
* Embedded software
* Web Applications
* Artificial Intelligence software
* Scientific software.

### **2) What is the main difference between a computer program and computer software?**

A computer program is a piece of programming code. It performs a well-defined task. On the other hand, the software includes programming code, documentation and user guide.

### **3) What is software re-engineering?**

It is a process of software development which is done to improve the maintainability of a software system.



### **4) Describe the software development process in brief:**

The software development is a life cycle is composed of the following stages:

* Requirement analysis
* Specification
* Software architecture
* Implementation
* Testing
* Documentation
* Training and support
* Maintenance

### **5) What are SDLC models available?**

Waterfall Model, Spiral Model, Big-bag model, Iterative Model, and V- Model are some of the famous [SDLC models](https://www.guru99.com/software-development-life-cycle-tutorial.html).

### **6) What is verification and validation?**

****Verification:****

Verification is a term that refers to the set of activities which ensure that software implements a specific function.

****Validation:****

It refers to the set of activities which ensure that software that has been built according to the need of clients.

### **7) In software development process what is the meaning of debugging?**

Debugging is the process that results in the removal of error. It is very important part of the successful testing.

### **8) How can you make sure that your code is both safe and fast?**

In the software, development security is always first. So if the execution of the program is slow then, I will try to identify the reason out ways to its time complexity.

### **9) Name two tools which are used for keeping track of software requirements?**

There many l ways to keep track of requirements.

****Two commonly used are:****

* Make a requirements specifications document to list all of the requirements.
* Create an excel sheet the list down the requirement, type, dependency, priority, etc.

### **10) What is the main difference between a stubs, a mock?**

A stub is a minimal implementation of an interface which generally returns hardcoded data while mock usually verifies outputs against expectations. Those expectations are set in the test.

### **11) What language do you like to write programming algorithms?**

Every developer has their views when it comes to the programming language choices. Though, one should prefer high-level languages because they are dynamic. Like C and C++ languages.

### **12) What is computer software?**

Computer software is a package which includes a software program, its documentation, and user guide on how to use the software.

### **13) According to you which SDLC model is the best?**

There, is no such ranking, as SDLC Models are adopted as per the need for the development process. It may differ software-to-software.

### **14) Who is software project manager? What is his role?**

A software project manager is a person responsible for managing the software development project.

The project manager is doing the project planning, monitoring the progress, communication. He or she also manages risks and resources to deliver the project within time, cost, and quality constraints.

### **15) What is mean by software scope?**

Software scope is a well-defined boundary. It includes all kind of activities that are done to develop and deliver the software product.

The software scope defines all functionalities and artifacts to be delivered as a part of the software. The scope also identifies what the product will do? What is not the part of the project? What is project estimation?

This process is helpful to estimate various aspects of the software product. This estimation can be decided either consulting experts or by using pre-defined formulas.

### **16) How to find the size of a software product?**

The size of software product can be calculated using by following two methods

* Counting the lines of delivered code
* Counting delivered function points

### **17) What are function points?**

Function points are the features which are provided by the software product. It is considered as a most important measurement for software size.

### **18) What are software project estimation techniques available?**

Most widely used estimation techniques are:

* Decomposition technique
* Empirical technique

### **19) What is Software configuration management?**

Software configuration management is a process of tracking and controlling changes that happen in the software.

Change control is a function which ensures that all changes made into the software system are consistent and created using organizational rules and regulations.

### **20) How can you measure project execution?**

We can measure project execution using Activity Monitoring, Status Reports, and Milestone Checklists.

### **21) Tell me about some project management tools.**

There are many types of management tools used as per the need for a software project. Some of them are Pert Chart, Gantt Chart, Resource Histogram, Status Reports, etc.

### **22) What are software requirements?**

Software requirements are a functional description of a proposed software system. It is assumed to be the description of the target system, its functionalities, and features.

### **23) What is feasibility study?**

It is a measure to find out how practical and beneficial the software project development will prove to the organization. The software analyzer conducts a study to know the economic, technical and operational feasibility of the project.

1. ****Economic:****It includes the cost of training, cost of additional and tools and overall estimation of costs and benefits of the project.
2. ****Technical:****It evaluate technical aspect. Is it possible to develop this system? Assessing the suitability of machine(s) and OS on which software will execute, knowledge of the software development and tools available for this project.
3. ****Operational:****Here the analyst need to assess that the organization will able to adjust smoothly to the changes done as per the demand for the project. Is the problem worth solving at the estimated cost?

After, studying all this the final feasibility report is created.

### **24) What are functional and non-functional requirements?**

Functional requirements are functional features which are expected by users from the proposed software product.

Non-functional requirements are related to security, performance, look, and feel of the user interface.

### **25) What is software metric?**

Software Metrics offers measures for various aspects of software process which are divided into:

1. Requirement metrics: Length requirements, completeness
2. Product metrics: Number of coding Lines, Object-oriented metrics, design and test metrics.

### **26) What is modularization?**

Modularization is a technique which is used for dividing a software system into various discreet modules. That is expected to carry out the tasks independently.

### **27) What is cohesion?**

Cohesion is a measure that defines the intra-dependability among the elements of the module.

### **28) Mentions some software analysis & design tools?**

Some of the most important software analysis and designing tools are:

* Data Flow Diagrams
* Structured Charts
* Structured English
* Data Dictionary
* Hierarchical Input Process Output diagrams
* Entity Relationship Diagrams and Decision tables

### **29) What is mean by level-0 Data flow diagram?**

Highest abstraction level is called Level 0 of DFD. It is also called context level DFD. It portrays the entire information system as one diagram.

### **30) What is the major difference between structured English and Pseudo Code?**

Structured English is native English language. It is used to write the structure of a program module. It uses programming language keywords. On the other hand, Pseudo Code is more like to the programming language without syntax of any specific language.

### **31) What is structured design?**

Structured design is a conceptualization of problem. It also called solution design and which is based on ‘divide and conquer’ strategy.

### **32) What is functional programming?**

It is a programming method, which uses the concepts of a mathematical function. It provides means of computation as mathematical functions, which also produces results irrespective of program state.

### **33) What is Quality Assurance vs. Quality Control?**

Quality Assurance checks if proper process is followed while developing the software while Quality Control deals with maintaining the quality of software product.

### **34) What are CASE tools?**

CASE means Computer Aided [Software Engineering](https://www.guru99.com/what-is-software-engineering.html). They are set of automated software application programs, which are used to support, enhance and strengthen the SDLC activities.

### **35) Which process model removes defects before software get into trouble?**

Clean room software engineering method removes defects before software gets into trouble.

### **36) Solve this problem**

There are twenty different socks of two types in a drawer in one dark room. What is the minimum number of socks you need to take to ensure you have a matching pair?”

If you pick up three socks, they may be of the same type even if the odds are 50%. Odds never an equal reality. Therefore, the only way to ‘ensure you have a matching pair’ is to pick up at least 11 number of shocks.

### **37) How you can make sure that your written code which can handle various kinds of error situation?**

I can write tests that define the expected error situations.

### **38) Explain the differences between a Thread and a Process?**

A process is instance of the computer program.In a single program it is possible to have one or more threads.

### **39) Tell me the difference between an EXE and a DLL?**

An exe is an executable program while a DLL is a file that can be loaded and executed by programs dynamically. It is an external code repository for programs. As both are different programs, reuse the same DLL instead of having that code in their file. It also reduces required storage space.

### **40) What is strong-typing and weak-typing? Which is preferred? Why?**

Strong typing checks the types of variables at compile time. On the other hand, weak typing checks the types of the system at run-time. Among them, Strong typing is always preferred because it minimizes the bugs.

### **41) Describe the difference between Interface-oriented, Object-oriented and Aspect-oriented programming.**

* Interface programming is contract based.
* Object-oriented is a way to write granular objects which have a single purpose.
* Aspect Oriented Programming is to segregate the code in such a manner that various objects carry the main tasks, and the subsidiary tasks are carried by independent objects.

### **42) Why using catch (exception) is always a bad idea?**

It is a bad idea because:

* As there is no variable defined, it is not possible to read the exception
* It’s good to use an exception when you have known exception types.

### **43) What type of data is passed via HTTP Headers?**

Script and metadata passed via HTTP headers.

### **44) How do you prioritize requirements?**

First, you need to design a system by evaluating data structure. Then you should move on to the code structure needed to support it.

### **45) Give me differences between object-oriented and component-based design?**

Object-oriented design can easily be encapsulated to some degree in component-based design.

### **46) When do you use polymorphism?**

Polymorphism is used when there is a need for override functionality when inheriting class. It’s about shared classes and shared contracts.

### **47) What is the difference between stack and queue?**

* Queue is always First In, First Out
* Stack is always Last In, First Out

### **48) What is essential for testing the quality of the code?**

According to me, the unit testing framework is essential for testing the quality of the code.

### **49) Do you think that the maintenance of software is expensive?**

According to me, maintenances of software will never be expensive if we are using proper development process.

### **50) Give me differences between tags and branches?**

Tags are for versioning releases which are temporary holding places for doing such thing. However, branches are deleted when those changes are merged into the trunk.

### **51) Where is a protected class-level variable available?**

Protected class-level variables are available to any sub-class derived from the base class.

### **52) Is it possible to execute multiple catch blocks for a single try statement?**

Yes. Multiple catch blocks can be executed for a single try statement.

### **53) When do you need to declare a class as abstract?**

We should declare a class as abstract in the following situations:

1. When the class is inherited from an abstract class, but not all the abstract methods have been overridden.
2. In the case when minimum one of the methods in the class is declared as an abstract.

### 1) What is Software Engineering?

Software engineering is defined as the function of the systematic, disciplined, quantified approach to the development, operations, and maintenance of software.

[Click here for more information](https://www.javatpoint.com/software-engineering-tutorial)

### 2) What are the elements to be considered in the System Model Construction?

Elements to be considered in the System Model Construction are:

* Assumption
* Simplification
* Limitation
* Constraints
* Preferences

### 3) What does a System Engineering Model accomplish?

System Engineering Model accomplishes the following:

* Define Processes that serve needs of view
* Represent behavior of process and assumption
* Explicitly define Exogenous and Endogenous Input
* It represents all Linkages that enable an engineer to understand aspect better.

### 4) Define Framework.

A framework is the Code Skeleton that can be fleshed out with particular classes or functionality and designed to address the specific problem at hand.

### 5) What are the characteristics of the software?

Characteristics of the software are:

* Software is engineered, not manufactured.
* Software does not wear out.
* Most software is custom-built rather than being assembled from components.

### 6) What are the various categories of software?

The various categories of software are:

* System software Application.
* Software Engineering / Scientific.
* Software Embedded software.
* Web Applications.
* Artificial Intelligence software.

### 7) What are the challenges in software?

The challenges in the software are:

* Copying with legacy systems.
* Heterogeneity challenge.
* Delivery times challenge.

### 8) Define Software process.

A software process is defined as the structured set of activities that are required to develop the software system.

[Click here for more information](https://www.javatpoint.com/software-processes)

### 9) What are the internal milestones?

They are the significant and quantifiable attributes of progress. They are the standard methods in the project which provide that we are on the right track. They are under the authority of the project manager.

### 10) What is the limitation of RAD Model?

Limitation of RAD Model are:

* It requires a sufficient number of Human Resources to create enough number of teams.
* Developers and Users are not committed,the system fails.
* It is not Properly Modularized building component may be Problematic.
* It is not applicable when there is more possibility for Technical Risk.

[Click here for more information](https://www.javatpoint.com/software-engineering-rapid-application-development-model)

### 11) What are the disadvantages of classic life cycle model?

Disadvantages of the classic life cycle model are:

* Real projects rarely follow the sequential flow. Iteration always occurs and creates a problem.
* Challenging for the customer to state all requirements.
* The working version of the program is not available. So the customer must have patience.

### 12) What are the merits of the incremental model?

The merits of the incremental model are:

* The incremental model can be accepted when there is less number of people include in the project.
* Technical risks can be handle with each increment.
* For a minimal period, at least the core product can be delivered to the user.

### 13) What is the disadvantage of the spiral model?

The disadvantage of the spiral model are:

1. It is based on user communication. If the interface is not proper, then the software product which gets created will not be the up to the mark.
2. It demands a vast risk assessment. If the risk assessment is completed correctly, then only the successful product can be obtained.

### 14) Name the Evolutionary process Models.

Evolutionary powers models are:

* Incremental model
* Spiral model
* WIN-WIN spiral model
* Concurrent Development

### 15) Define Software Prototyping.

Software prototyping is represented as rapid software development for validating the requirements.

### 16) What are the benefits of prototyping?

The benefits of prototyping are:

* Prototype services as a basis for developing system specification.
* Design quality can be revised.
* The system can be managed easily.
* Development efforts may get decreased.
* System usability can be upgraded.

### 17) What are the prototyping methods in software process?

The prototyping methods in the software process are:

* ****Evolutionary prototyping:**** In this method of system development, the initial prototype is arranged, and it is then precise through the number of phases to the final stage.
* ****Throw-away prototyping:**** Using this method, a rough practical implementation of the system is produced. The requirement issues can be identified from this implementation. It is then rejected. System is then developed using some various engineering paradigm.

### 18) What are the advantages of evolutionary prototyping?

The advantages of evolutionary prototyping are:

* Fast delivery of the working system.
* User is contained while developing the system.
* The more useful system can be delivered.
* Specification, design and implementation work in equivalent manner.

### 19) What are the various Rapid prototyping techniques?

The various rapid prototyping techniques are:

* Dynamic high-level language development.
* Database programming.
* Component and application assembly.

### 20) What are the uses of User-Interface Prototyping?

This prototyping is used to pre-specify the looks and effectively feel of customer interface.

### 21) What is the principle of the prototype model?

A prototype is built to quickly determine to the user what the product would look like. The only minimal functionality of the actual product is supported during the prototyping phase.

### 22) Define System Context Diagram (SCD)?

System Context Diagram (SCD):

* Establish data boundary between System being implemented and Environment in which system operates.
* Describes all external producers, external consumers, and entities that communicate through the customer interface.

### 23) Define Quality Function Deployment (QFD)?

Quality Function Deployment (QFD) is a method that translates the needs of the user into a technical requirement. It concentrates on maximizing user satisfaction from the software engineering process.

### 24) What is Requirement Engineering?

Requirement engineering is the process of establishing services which the user required from the system and constraint under which it operates and is developed.

[Click here for more information](https://www.javatpoint.com/software-engineering-requirement-engineering)

### 25) What is ERD?

Entity Relationship Diagram is the graphical description of the object relationship pair. It is primarily used in the database application.

### 26) What is DFD?

Data Flow Diagram depicts the data flow and the transforms which are applied to the data as it moves from input to output.

### 27) What is a state transition diagram?

State transition diagram is a collection of states and events. The events cause the operation to change its state. It also describes what actions are to be taken on the occurrence of particular events.

### 28) What is Software Quality Assurance?

Software Quality Assurance is a set of auditing and documenting functions that assess the effectiveness and completeness of quality control activities.

[Click here for more information](https://www.javatpoint.com/software-quality-assurance)

### 29) What is the use of CMM?

Software Quality means Conformance to state functional explicitly and performance requirements, explicitly documented development standards, inherent characteristics expected for professionally developed software.

### 30) What is coupling?

Coupling is the significant measure of the degree to which classes are linked to one another. Coupling should be kept as low as possible.

### 31) What is cohesion?

Cohesion is the indication of the relative functional strength of a module. It is a natural extension of Information Hiding and Performs a single task, requiring little integration with other components.

[Click here for more information](https://www.javatpoint.com/software-engineering-coupling-and-cohesion)

### 32) Define Refactoring.

Refactoring means changing a software system in a way that does not alter the external behavior of code.

### 33) What is Software Architecture?

Software Architecture means the overall structure of the software and how that software provides conceptual integrity for the system.

### 34) Define Stamp coupling.

When a portion of the data structure is passed via the module interface, then it is called as stamp coupling.

### 35) Define common coupling.

When several modules reference a global data area, then the coupling is called common coupling.

### 36) Define temporal cohesion.

When a module contains tasks that are related by the fact that all must be executed within the same period, then it is termed as temporal cohesion.

### 37) Define metrics.

Metrics are defined as the degree to which a system component or process possesses a given attribute.

### 38) What is COCOMO model?

Constructive Cost Model is a cost model, which gives the estimate of several staff-months it will take to develop the software product.

[Click here for more information](https://www.javatpoint.com/cocomo-model)

### 39) What is the purpose of the timeline chart?

The objective of the timeline chart is to emphasize the scope of the individual task. Hence set of functions are given as input to the timeline chart.

### 40) Define Smoke Testing?

Smoke testing is Integration Testing and frequently used when software products are being developed.

### 41) What are the benefits of Smoke Testing?

Benefits of doing Smoke Testing are:

* Integration Risk is minimized.
* Quality of end-product is improved.
* Error diagnosis and Correction are simplified.
* Progress is easy to assess.

### 42) What is Equivalence Partition?

Equivalence Partitions Derives an input domain of a program into classes of data from which test cases are derived. It is a Set of Objects have linked by relationships as Symmetric, Transitive, and Reflexive an equivalence class is present.

### 43) What are the steps followed in testing?

The steps followed in testing are:

* ****Unit testing:**** The individual elements are tested in this type of testing.
* ****Module testing:**** Related group of independent items is tested.
* ****Sub-system testing:**** This is a type of integration testing. Different modules are integrated into a sub-system, and the entire subsystem is tested.
* ****System testing:**** The entire system is tested in this system.
* ****Acceptance testing:**** This type of testing contains testing of the system with user data if the system behaves as per client need, then it is accepted.

### 44) Distinguish between Alpha and Beta testing.

Alpha and Beta testings are the two types of acceptance testing.

* ****Alpha test:**** The alpha testing is attesting in which the customer tests the version of complete software under the supervision of the developer. This testing is implement at the developer's site.
* ****Beta test:**** The beta testing is a testing in which the customer tests the version of the software without the developer being present. This testing is performed at the customer's site.

### 45) What are the types of Static Testing tools?

There are the three types of static testing tools.

* ****Code-based testing tools:**** These tools take source code as input and generate test cases.
* ****Specialized testing tools:**** Using this language, the detailed test specification can be written for each test case.
* ****Requirement-based testing tools:**** These tools help in designing as per user requirements.

### 46) Define maintenance.

Maintenance is described as the process in which changes are implemented by either modifying the existing system?s architecture or by adding new components to the system.

[Click here for more information](https://www.javatpoint.com/software-engineering-software-maintenance)

### 47) What are the types of software maintenance?

Types of software maintenance are:

****Corrective Maintenance:**** It means the maintenance for correcting the software faults.

****Adaptive maintenance:**** It means maintenance for adapting the change in environment.

****Perfective maintenance:**** It means modifying or enhancing the system to meet the new requirements.

****Preventive maintenance:**** It means changes made to improve future maintainability.

[Click here for more information](https://www.javatpoint.com/software-engineering-causes-of-software-maintenance-problems)

### 48) What is CASE Tools?

CASE Tools stands for Computer-Aided Software Engineering. It is system software that provides automated support for software process activities. It contains program used to support software process operations such as Requirement Analysis, System Modeling. Debugging and Testing.

[Click here for more information](https://www.javatpoint.com/software-engineering-case-tools-for-software-metrics)

### 49) What is Risk management?

Risk management is the phase of anticipating hurdles in carrying out the original plan and providing alternate methods so that the impact on the anticipated initially outcome is minimal.

**Q.What is computer software?**

**A.** Computer software is a complete package, which includes software program, its documentation and user guide on how to use the software.

**Q.Can you differentiate computer software and computer program?**

**A.** A computer program is piece of programming code which performs a well defined task where as software includes programming code, its documentation and user guide.

**Q.What is software engineering?**

**A.** Software engineering is an engineering branch associated with software system development.

**Q.When you know programming, what is the need to learn software engineering concepts?**

**A.** A person who knows how to build a wall may not be good at building an entire house. Likewise, a person who can write programs may not have knowledge of other concepts of Software Engineering. The software engineering concepts guide programmers on how to assess requirements of end user, design the algorithms before actual coding starts, create programs by coding, testing the code and its documentation.

**Q.What is software process or Software Development Life Cycle (SDLC)?**

**A.**Software Development Life Cycle, or software process is the systematic development of software by following every stage in the development process namely, Requirement Gathering, System Analysis, Design, Coding, Testing, Maintenance and Documentation in that order.

**Q.What are SDLC models available?**

**A.** There are several SDLC models available such as Waterfall Model, Iterative Model, Spiral model, V-model and Big-bang Model etc.

**Q.What are various phases of SDLC?**

**A.** The generic phases of SDLC are: Requirement Gathering, System Analysis and Design, Coding, Testing and implementation. The phases depend upon the model we choose to develop software.

**Q.Which SDLC model is the best?**

**A.** SDLC Models are adopted as per requirements of development process. It may very software-to-software to ensuring which model is suitable.

We can select the best SDLC model if following answers are satisfied -

* Is SDLC suitable for selected technology to implement the software ?
* Is SDLC appropriate for client’s requirements and priorities ?
* Is SDLC model suitable for size and complexity of the software ?
* Is the SDLC model suitable for type of projects and engineering we do ?
* Is the SDLC appropriate for the geographically co-located or dispersed developers ?

**Q.What is software project management?**

**A.** Software project management is process of managing all activities like time, cost and quality management involved in software development.

**Q.Who is software project manager?**

**A.**A software project manager is a person who undertakes the responsibility of carrying out the software project.

**Q.What does software project manager do?**

**A.**Software project manager is engaged with software management activities. He is responsible for project planning, monitoring the progress, communication among stakeholders, managing risks and resources, smooth execution of development and delivering the project within time, cost and quality contraints.

**Q.What is software scope?**

**A.** Software scope is a well-defined boundary, which encompasses all the activities that are done to develop and deliver the software product.

The software scope clearly defines all functionalities and artifacts to be delivered as a part of the software. The scope identifies what the product will do and what it will not do, what the end product will contain and what it will not contain.

**Q.What is project estimation?**

**A.** It is a process to estimate various aspects of software product in order to calculate the cost of development in terms of efforts, time and resources. This estimation can be derived from past experience, by consulting experts or by using pre-defined formulas.

**Q.How can we derive the size of software product?**

**A.** Size of software product can be calculated using either of two methods -

* Counting the lines of delivered code
* Counting delivered function points

**Q.What are function points?**

**A.** Function points are the various features provided by the software product. It is considered as a unit of measurement for software size.

**Q.What are software project estimation techniques available?**

**A.** There are many estimation techniques available.The most widely used are -

* Decomposition technique (Counting Lines of Code and Function Points)
* Empirical technique (Putnam and COCOMO).

**Q.What is baseline?**

**A.** Baseline is a measurement that defines completeness of a phase. After all activities associated with a particular phase are accomplished, the phase is complete and acts as a baseline for next phase.

**Q.What is Software configuration management?**

**A.** Software Configuration management is a process of tracking and controlling the changes in software in terms of the requirements, design, functions and development of the product.

**Q.What is change control?**

**A.** Change control is function of configuration management, which ensures that all changes made to software system are consistent and made as per organizational rules and regulations.

**Q.How can you measure project execution?**

**A.** We can measure project execution by means of Activity Monitoring, Status Reports and Milestone Checklists.

**Q.Mention some project management tools.**

**A.** There are various project management tools used as per the requirements of software project and organization policies. They include Gantt Chart, PERT Chart, Resource Histogram, Critical Path Analysis, Status Reports, Milestone Checklists etc.

**Q.What are software requirements?**

**A.** Software requirements are functional description of proposed software system. Requirements are assumed to be the description of target system, its functionalities and features. Requirements convey the expectations of users from the system.

**Q.What is feasibility study?**

**A.** It is a measure to assess how practical and beneficial the software project development will be for an organization. The software analyzer conducts a thorough study to understand economic, technical and operational feasibility of the project.

**Economic**- Resource transportation, cost for training, cost of additional utilities and tools and overall estimation of costs and benefits of the project.

**Technical**- Is it possible to develop this system ? Assessing suitability of machine(s) and operating system(s) on which software will execute, existing developers’ knowledge and skills, training, utilities or tools for project.

**Operational**- Can the organization adjust smoothly to the changes done as per the demand of project ? Is the problem worth solving ?

**Q.How can you gather requirements?**

**A.** Requirements can be gathered from users via interviews, surveys, task analysis, brainstorming, domain analysis, prototyping, studying existing usable version of software, and by observation.

**Q.What is SRS?**

**A.** SRS or Software Requirement Specification is a document produced at the time of requirement gathering process. It can be also seen as a process of refining requirements and documenting them.

**Q.What are functional requirements?**

**A.** Functional requirements are functional features and specifications expected by users from the proposed software product.

**Q.What are non-functional requirements?**

**A.** Non-functional requirements are implicit and are related to security, performance, look and feel of user interface, interoperability, cost etc.

**Q.What is software measure?**

**A.** Software Measures can be understood as a process of quantifying and symbolizing various attributes and aspects of software.

**Q.What is software metric?**

**A.** Software Metrics provide measures for various aspects of software process and software product. They are divided into –

* Requirement metrics : Length requirements, completeness
* Product metrics :Lines of Code, Object oriented metrics, design and test metrics
* Process metrics: Evaluate and track budget, schedule, human resource.

**Q.What is modularization?**

**A.** Modularization is a technique to divide a software system into multiple discreet modules, which are expected to carry out task(s) independently.

**Q.What is concurrency and how it is achieved in software?**

**A.** Concurrency is the tendency of events or actions to happen simultaneously. In software, when two or more processes execute simultaneously, they are called concurrent processes.

### Example

While you initiate print command and printing starts, you can open a new application.

Concurrency, is implemented by splitting the software into multiple independent units of execution namely processes and threads, and executing them in parallel.

**Q.What is cohesion?**

**A.** Cohesion is a measure that defines the degree of intra-dependability among the elements of the module.

**Q.What is coupling?**

**A.** Coupling is a measure that defines the level of inter-dependability among modules of a program.

**Q.Mentions some software analysis & design tools?**

**A.** These can be: DFDs (Data Flow Diagrams), Structured Charts, Structured English, Data Dictionary, HIPO (Hierarchical Input Process Output) diagrams, ER (Entity Relationship) Diagrams and Decision tables.

**Q.What is level-0 DFD?**

**A.** Highest abstraction level DFD is known as Level 0 DFD also called a context level DFD, which depicts the entire information system as one diagram concealing all the underlying details.

**Q.What is the difference between structured English and Pseudo Code?**

**A.** Structured English is native English language used to write the structure of a program module by using programming language keywords, whereas, Pseudo Code is more close to programming language and uses native English language words or sentences to write parts of code.

**Q.What is data dictionary?**

**A.** Data dictionary is referred to as meta-data. Meaning, it is a repository of data about data. Data dictionary is used to organize the names and their references used in system such as objects and files along with their naming conventions.

**Q.What is structured design?**

**A.** Structured design is a conceptualization of problem into several well-organized elements of solution. It is concern with the solution design and based on ‘divide and conquer’ strategy.

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

**A.** Function-oriented design is comprised of many smaller sub-systems known as functions. Each function is capable of performing significant task in the system. Object oriented design works around the real world objects (entities), their classes (categories) and methods operating on objects (functions).

**Q.Briefly define top-down and bottom-up design model.**

**A.** Top-down model starts with generalized view of system and decomposes it to more specific ones, whereas bottom-up model starts with most specific and basic components first and keeps composing the components to get higher level of abstraction.

**Q.What is the basis of Halstead’s complexity measure?**

**A.** Halstead’s complexity measure depends up on the actual implementation of the program and it considers tokens used in the program as basis of measure.

**Q.Mention the formula to calculate Cyclomatic complexity of a program?**

**A.** Cyclomatic complexity uses graph theory’s formula: V(G) = e – n + 2

**Q.What is functional programming?**

**A.** Functional programming is style of programming language, which uses the concepts of mathematical function. It provides means of computation as mathematical functions, which produces results irrespective of program state.

**Q.Differentiate validation and verification?**

**A.** Validation checks if the product is made as per user requirements whereas verification checks if proper steps are followed to develop the product.

Validation confirms the right product and verification confirms if the product is built in a right way.

**Q.What is black-box and white-box testing?**

**A.** Black-box testing checks if the desired outputs are produced when valid input values are given. It does not verify the actual implementation of the program.

White-box testing not only checks for desired and valid output when valid input is provided but also it checks if the code is implemented correctly.

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Black Box Testing** | **White Box Testing** |
| Knowledge of software program, design and structure essential | No | Yes |
| Knowledge of Software Implementation essential | No | Yes |
| Who conducts this test on software | Software Testing Employee | Software Developer |
| baseline reference for tester | Requirements specifications | Design and structure details |

**Q.Quality assurance vs. Quality Control?**

**A.** Quality Assurance monitors to check if proper process is followed while software developing the software.

Quality Control deals with maintaining the quality of software product.

**Q.What are various types of software maintenance?**

**A.** Maintenance types are: corrective, adaptive, perfective and preventive.

* **Corrective**

Removing errors spotted by users

* **Adaptive**

tackling the changes in the hardware and software environment where the software works

* **Perfective maintenance**

implementing changes in existing or new requirements of user

* **Preventive maintenance**

taking appropriate measures to avoid future problems

**Q.What is software re-engineering?**

**A.** Software re-engineering is process to upgrade the technology on which the software is built without changing the functionality of the software. This is done in order to keep the software tuned with the latest technology.

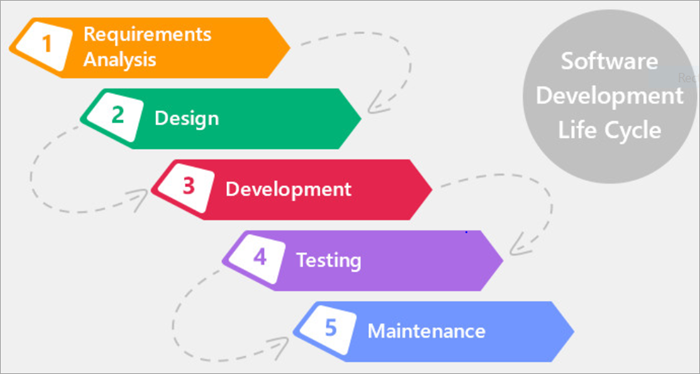
**Q.What are CASE tools?**

**A.** CASE stands for Computer Aided Software Engineering. CASE tools are set of automated software application programs, which are used to support, accelerate and smoothen the SDLC activities.

****Q #1) What is SDLC?****

****Answer:**** SDLC stands for Software Development Life Cycle. It defines the step by step approach for the development of software. [SDLC](https://www.softwaretestinghelp.com/software-development-life-cycle-sdlc/) involves the following phases i.e. Requirement Gathering, System Analysis, Design, Coding, Testing, Maintenance, and Documentation.

****Given below is the high-level representation of the various phases involved in SDLC.****

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2019/07/SDLC.png)

*[image [source](https://www.linkedin.com/" \t "https://www.softwaretestinghelp.com/software-engineering-interview-questions/_blank) ]*

****Q #2) What are the various models available in SDLC?****

****Answer:****There are several models available in SDLC for efficiently carrying out software development. Some of the models include the [Waterfall model](https://www.softwaretestinghelp.com/what-is-sdlc-waterfall-model/), V-Model, Agile model, etc.

****Q #3) Explain the term Baseline.****

****Answer:****A baseline is a milestone on the project which is usually defined by the project manager. Baselines are used to track the progress of the project from time to time to assess the overall health of the project.

****Q #4) What are the responsibilities of a Software Project Manager?****

****Answer:****A Software Project Manager is responsible for driving the project towards successful completion. It is the responsibility of the Software Project Manager to make sure the entire team follows a systematic and well-defined approach towards the development of software.

****A software project manager is also responsible for the following tasks:****

* Project planning
* Project status tracking
* Resource management
* Risk management
* Project delivery within time and budget.

****Q #5) What is Cohesion?****

****Answer:****Cohesion is the degree to which the elements of a module are inter-related to one another. It is like an internal glue that binds the elements of a module together. Good software has high levels of cohesion.

****Q #6) What is Coupling?****

****Answer:****Coupling is the degree of interdependence between the modules. Good software has low levels of coupling.

****Q#7) Explain the concept of Modularization.****

****Answer:****Modularization is used to divide software into multiple components or modules. Each module is worked upon by an independent development and testing team. The final result would be to combine multiple modules into a single working component.

****Q #8) What is Software Configuration Management?****

****Answer:****Software configuration management is the process of tracking and controlling the changes that occur during the software development lifecycle. Any change made during software development has to be tracked through a well-defined and controlled process.

Configuration management ensures that any changes made during software development are being controlled through a well-defined process.

****Q #9) What are the various phases of SDLC?****

****Answer:****The following are the most common phases of SDLC.

* Requirement Analysis
* Design
* Coding
* Testing
* Maintenance

****Q #10) Provide examples of Project Management tools.****

****Answer:****Given below are some of the most commonly used project management tools that are available in the industry today.

* Gantt Chart
* Checklists
* Status Reports
* Histograms
* Microsoft Project

***Recommended Read =>*[Top Project Management Tools That You Should Know](https://www.softwaretestinghelp.com/project-management-software-tools/)**

****Q #11) What are CASE tools?****

****Answer:****CASE stands for Computer-Aided Software Engineering tools that are utilized to support and accelerate the various activities of the Software Development Lifecycle.

****Q #12) What is Black box testing?****

****Answer:****Black box testing involves testing the application without the knowledge of the internal structure or code implementation. Testers would only bother about the functionality of the software in black box testing rather than data flow and code execution in the back end.

****Q #13) What is White box testing?****

****Answer:****White box testing is testing the application with the knowledge of the internal structure and code implementation. This testing is generally performed by the developer who has written the code in the form of unit tests.

****Q #14) What is a Feasibility Study?****

****Answer:****A feasibility study is conducted on a software product to assess how practical and beneficial is the development of the software product to the organization. Software is analyzed thoroughly to understand the economic and technical aspects of a software product to be developed.

****Q #15) How can you measure Project execution?****

****Answer:****Project execution status can be monitored using the following techniques.

* Status Reports
* Milestone checklists
* Activity Monitoring

****Q #16) What are the Functional Requirements?****

****Answer:****Functional requirements are the features that a developed software product is expected to perform. ****For example,**** adding a payment option at an eCommerce website will be a functional requirement.

****Q #17) What are Non-Functional Requirements?****

****Answer:****Non-functional requirements measure the usability of the application such as User Interface look and feel, Security, Performance, Interoperability, Reliability, etc.

****Q #18) What is the difference between Quality Assurance and Quality Control?****

****Answer:****Quality Assurance is ensuring that the delivered software has the least number of defects possible. Quality Control is the process of ensuring that the quality of the product is maintained in the long run.

Quality Assurance is done by the testing team of the project while Quality Control is usually done by a dedicated support team, who is responsible for the quality of the product even if the product is under the maintenance phase of software engineering.

***Also, Read =>*[Quality Assurance Vs Quality Control](https://www.softwaretestinghelp.com/quality-assurance-vs-quality-control/)**

****Q #19) What is the difference between Verification and Validation?****

****Answer:****Verification is the process of ensuring that the product is built right, from a process and standards perspective.

Validation is the process of ensuring that we build the right product, from a customer perspective. Verification is a static testing methodology wherein the product is tested without executing the code, while validation is a dynamic testing methodology.

*****Worth Reading =>***[Complete Study of Verification and Validation](https://www.softwaretestinghelp.com/difference-between-verification-vs-validation/)**

****Q #20) Which SDLC model is the best to choose for a Software Product?****

****Answer:****There are no rules as such stating which specific SDLC model has to be used for a software product. It depends on the type of software project being built and the organization’s policies & procedures.

****Q #21) What do you mean by Software Scope?****

****Answer:****Software scope is the list of features provided by the developed software. Based on the scope of the software, estimations such as time allocation, budget and resource allocation can be done.

****Q #22) What is SRS?****

****Answer:****SRS stands for Software Requirement Specification (SRS) document. It is a document to capture all the functional and non-functional requirements of a product. Not all SDLC models need to follow [SRS documents](https://www.softwaretestinghelp.com/rview-srs-document-and-create-test-scenarios-software-testing-training-course-day-2/), some models capture requirements in the form of user stories, whereas some models in the form of excel sheets, etc.

****Q #23) What is the SDLC model that you have used in your previous project?****

****Answer:****The answer to this question depends on the experience of an interview candidate. If the candidate answers the SDLC model to be the Waterfall model, then the interviewer will start asking questions about the Waterfall model and if he answers it to be Agile, then the interviewer will start asking terms related to Agile methodology such as Scrum, Sprint, etc.

****Q #24) Explain the Waterfall model in detail.****

****Answer:****The ****waterfall**** model is a sequential model in which the next phase starts only after the first phase is completed. ****For example,**** the testing phase will start only after the development phase is complete, the maintenance phase will start only after the testing phase is complete.

****Below are the various phases involved in the waterfall model.**** Please note that the number of phases and sequences of phases may vary from one project to another.

* Requirements
* Design
* Coding
* Testing
* Maintenance

****a) Requirements:****This is the phase where the system to be developed is documented in the form of Software Requirement Specification (SRS) document. This is the most important phase of SDLC as a clear understanding of requirements from the client will reduce the rework in the following phases.

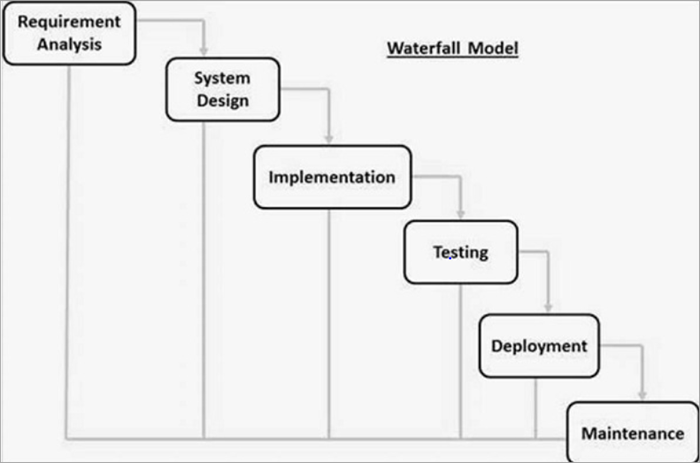
****b) Design:****This is the phase where the architecture of the system to be developed is finalized. Architecture can be in the form of a high-level design or a low-level design. Architecture must also include the hardware and software specifications of the system to be developed.

****c) Coding:****This is the phase where the code for the system to be developed is written. [Unit Testing](https://www.softwaretestinghelp.com/unit-testing/) and [Integration Testing](https://www.softwaretestinghelp.com/what-is-integration-testing/) must be performed by the developers at this stage before deploying the code for testing.

****d) Testing:****This is the phase where the product developed is tested by an independent testing team to validate if it meets the requirements in the Software Requirement Specification (SRS). Defects raised at this phase need to be fixed before providing sign off on the product.

****e) Maintenance:****This phase comes once the testing phase is complete. It takes care of any production issues that may arise after the product is delivered to the customer. The duration of the maintenance phase differs from project to project and one organization to another.

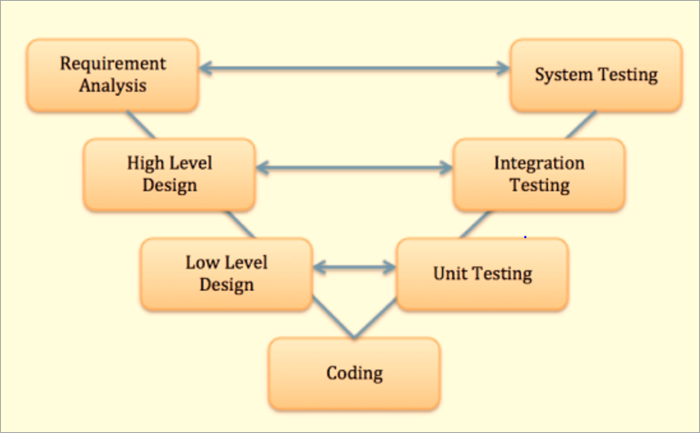
****Below is the diagram to depict the waterfall model in the form of phases.****

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2019/07/WaterfallModel.png)

****Q #25) Explain V-Model in detail.****

****Answer:****V-Model stands for the verification and validation model. [V-model](https://www.softwaretestinghelp.com/what-is-stlc-v-model/) is an addition to the waterfall model, in the sense that V-model is also a sequential model. In V-model, each phase of development is associated with a corresponding testing phase.

****The image given below depicts the various phases involved in V-model.****

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2019/07/V-Model.png)

The left side of the model is the Software Development Lifecycle while the right side of the model is Software Testing Lifecycle. As the phases form the shape of the letter ‘V’, this model is called V-Model.

****Explanation:****

Within the V-Model, SDLC is to be interpreted from top to bottom, while STLC is to be interpreted from the bottom to the top. Initially, requirements are gathered to document the system to be developed as per the client requirements. The testing team develops the system test plan based on the requirements.

Then comes the high-level design and the detailed level design phases where the architecture of the system is prepared. The testing team prepares the Integration Test plan in these phases. Once the coding is complete on SDLC, STLC will start from unit testing, followed by integration testing and System testing.

**1. What is software re-engineering?**

Software reengineering is the process of scanning, modifying, and reconfiguring a system in a new way. The principle of reengineering applied to the software development process is called software reengineering. It has a positive impact on software cost, quality, customer service, and shipping speed. Software reengineering improves software to create it more efficiently and effectively.

For more details please refer to [What Is Software Re-Engineering?.](https://www.geeksforgeeks.org/software-re-engineering/)

**2. What are the characteristics of Software?**

There are various characteristics of software:

* **Software is developed or engineered; it is not manufactured in the classical sense:**
  + Although some similarities exist between software development and hardware manufacturing, few activities are fundamentally different.
  + In both activities, high quality is achieved through good design, but the manufacturing phase for hardware can introduce quality problems than software.
* **The software doesn’t “wear out.”:**
  + Hardware components suffer from the growing effects of many other environmental factors. Stated simply, the hardware begins to wear out.
  + Software is not susceptible to the environmental maladies that cause hardware to wear out.
  + When a hardware component wears out, it is replaced by a spare part.
  + There are no software spare parts.
  + Every software failure indicates an error in design or in the process through which design was translated into machine-executable code. Therefore, the software maintenance tasks that accommodate requests for change involve considerably more complexity than hardware maintenance. However, the implication is clear—the software doesn’t wear out. But it does deteriorate.
* **The software continues to be custom-built:**
  + A software part should be planned and carried out with the goal that it tends to be reused in various projects.
  + Current reusable segments encapsulate the two information and the preparation that is applied to the information, empowering the programmer to make new applications from reusable parts.
  + In the hardware world, component reuse is a natural part of the engineering process

For more details please refer to the following article[Software Engineering Characteristics](https://www.geeksforgeeks.org/software-engineering-software-characteristics/).

**3. What activities come under the umbrella** **activities?**

The activities of the software engineering process framework are complemented by a variety of higher-level activities. Umbrella activities typically apply to the entire software project and help the software team manage and control progress, quality, changes, and risks. Common top activities include Software Project Tracking and Control Risk Management, Software Quality Assurance Technical Review Measurement Software Configuration Management Reusability Management Work Product Preparation and Production, etc.

**4. What is Cohesion and Coupling?**

Cohesion indicates the relative functional capacity of the module. Aggregation modules need to interact less with other sections of other parts of the program to perform a single task. It can be said that only one coagulation module (ideally) needs to be run. Cohesion is a measurement of the functional strength of a module. A module with high cohesion and low coupling is functionally independent of other modules. Here, functional independence means that a cohesive module performs a single operation or function. The coupling means the overall association between the modules.

Coupling relies on the information delivered through the interface with the complexity of the interface between the modules in which the reference to the section or module was created. High coupling support Low coupling modules assume that there are virtually no other modules. It is exceptionally relevant when both modules exchange a lot of information. The level of coupling between two modules depends on the complexity of the interface.

For more details, please refer to the following article [Coupling and cohesion.](https://www.geeksforgeeks.org/software-engineering-coupling-and-cohesion/)

**5. What are the various phases of SDLC?**

**SDLC phases:**

* Requirement gathering & analysis
* Design
* Implementation & coding
* Testing
* Deployment
* Maintenance

For more details, please refer to the following article [Software Development Life Cycle.](https://www.geeksforgeeks.org/software-development-life-cycle-sdlc/)

**6. What is the name of various CASE tools?**

* Requirement Analysis Tool
* Structure Analysis Tool
* Software Design Tool
* Code Generation Tool
* Test Case Generation Tool
* Document Production Tool
* Reverse Engineering Tool

For more details, please refer to the following article[Computer-Aided Software Engineering(CASE).](https://www.geeksforgeeks.org/computer-aided-software-engineering-case/)

**7. What is Black box testing?**

The black box test (also known as the conducted test closed box test opaque box test) is centered around software useful prerequisites. In other words, it is possible to guess a set of information conditions that help the program through an attempt to discover and fulfill all the necessities perfectly. There is no choice of black-box testing white box procedures. Maybe it’s a complementary methodology, perhaps the white box method will reveal the errors of other classes.

For more details, please refer to [the following article Software Engineering – Black Box Testing.](https://www.geeksforgeeks.org/software-engineering-black-box-testing/)

**8.  What is White box testing?**

White Box Testing is a method of analyzing the internal structure, data structures used, internal design, code structure, and behavior of software, as well as functions such as black-box testing. Also called glass-box test or clear box test or structural test.

For more details, please refer to the following article[Software Engineering – White Box Testing.](https://www.geeksforgeeks.org/software-engineering-white-box-testing/)

**9.  What is a Feasibility Study?**

The Feasibility Study in Software Engineering is a study to assess the adequacy of proposed projects and systems. A feasibility study is a measure of a software product on how product development can benefit an organization from a validity analysis or practical point of view. Feasibility studies are conducted for multiple purposes to analyze the correctness of a software product in terms of development, porting, the contribution of an organization’s projects, and so on.

For more details, please refer to the following article [Types of Feasibility Study in Software Project Development article.](https://www.geeksforgeeks.org/types-of-feasibility-study-in-software-project-development/)

**10.  What is the Difference Between Quality Assurance and Quality Control?**

| Quality Assurance (QA) | Quality Control (QC) |
| --- | --- |
| It focuses on providing assurance that the quality requested will be achieved. | It focuses on fulfilling the quality requested. |
| It is the technique of managing quality. | It is the technique to verify quality. |
| It does not include the execution of the program. | It always includes the execution of the program. |
| It is a managerial tool. | It is a corrective tool. |
| It is process-oriented. | It is product-oriented. |
| The aim of quality assurance is to prevent defects. | The aim of quality control is to identify and improve the defects. |
| It is a preventive technique. | It is a corrective technique. |
| It is a proactive measure. | It is a reactive measure. |
| It is responsible for the full software development life cycle. | It is responsible for the software testing life cycle. |
| Example: Verification | Example: Validation |

**11.  What is the difference between Verification and Validation?**

| Verification | Validation |
| --- | --- |
| Verification is a static practice of verifying documents, design, code, black-box, and programs human-based. | Validation is a dynamic mechanism of validation and testing the actual product. |
| It does not involve executing the code. | It always involves executing the code. |
| It is human-based checking of documents and files. | It is computer-based execution of the program. |
| Verification uses methods like inspections, reviews, walkthroughs, and Desk-checking, etc. | Validation uses methods like black box (functional) testing, gray box testing, and white box (structural) testing, etc. |
| Verification is to check whether the software conforms to specifications. | Validation is to check whether the software meets the customer’s expectations and requirements. |
| It can catch errors that validation cannot catch. | It can catch errors that verification cannot catch. |
| Target is requirements specification, application and software architecture, high level, complete design, and database design, etc. | Target is an actual product-a unit, a module, a bent of integrated modules, and an effective final product. |
| Verification is done by QA team to ensure that the software is as per the specifications in the SRS document. | Validation is carried out with the involvement of the testing team |
| It generally comes first done before validation. | It generally follows after verification. |
| It is low-level exercise. | It is a High-Level Exercise. |

For more details, please refer to the following article [Software Engineering – Verification and Validation.](https://www.geeksforgeeks.org/software-engineering-verification-and-validation/)

**12. What is reverse engineering?**

**Software Reverse Engineering** is a process of recovering the design, requirement specifications, and functions of a product from an analysis of its code. It builds a program database and generates information from this. The purpose of reverse engineering is to facilitate maintenance work by improving the understandability of a system and producing the necessary documents for a legacy system.

**Reverse Engineering Goals:**

* Cope with Complexity.
* Recover lost information.
* Detect side effects.
* Synthesize higher abstraction.
* Facilitate Reuse.

For more details, please refer to the following article [Software Engineering – Reverse Engineering.](https://www.geeksforgeeks.org/software-engineering-reverse-engineering/)

**13. What is SRS?**

**Software Requirement Specification (SRS) Format** is a complete specification and description of requirements of the software that needs to be fulfilled for successful development of software system. These requirements can be functional as well as non-requirements depending upon the type of requirement. The interaction between different customers and contractors is done because it’s necessary to fully understand the needs of customers. For more details please refer[software requirement specification format article.](https://www.geeksforgeeks.org/software-requirement-specification-srs-format/)

**14. Distinguish between Alpha and Beta testing.**

| Alpha Testing | Beta Testing |
| --- | --- |
| Alpha testing involves both white box and black box testing. | Beta testing commonly uses black-box testing. |
| Alpha testing is performed by testers who are usually black,it -box internal employees of the organization. | Beta testing is performed by clients who are not part of the organization. |
| Alpha testing is performed at the developer’s site. | Beta testing is performed at the end-user,  the of the product. |
| Reliability and security testing are not checked in alpha testing. | Reliability, security, and robustness are checked during beta testing. |
| Alpha testing ensures the quality of the product before forwarding it to beta testing. | Beta testing also concentrates on the quality of the product but collects the user’s time-long input on the product and ensures that the product is ready for real-time users. |
| Alpha testing requires a testing environment or a lab. | Beta testing doesn’t require a testing environment or lab. |
| Alpha testing may require a real-time long execution cycle. | Beta testing requires only a few weeks of execution. |
| Developers can immediately address the critical issues or fixes in alpha testing. | Most of the issues or feedback collected from the beta testing will be implemented in future versions of the product |

For more details, please refer to the following article[Alpha Testing](https://www.geeksforgeeks.org/alpha-testing-software-testing/) and [Beta Testing](https://www.geeksforgeeks.org/beta-testing-software-testing/).

**15.  What are the elements to be considered in the System Model Construction?**

The type and size of the software, the experience of use for reference to predecessors, difficulty level to obtain users’ needs, development techniques and tools, the situation of the development team, development risks, the software development methods should be kept in mind. It is an important prerequisite to ensure the success of software development that designing a reasonable and suitable software development plan.

**16. What are CASE tools?**

CASE stands for Computer-Aided Software Engineering. CASE tools are a set of automated software application programs, which are used to support, accelerate and smoothen the SDLC activities.

**17.  What is the limitation of the RAD Model?**

* For large but scalable projects RAD requires sufficient human resources.
* Projects fail if developers and customers are not committed in a much-shortened time frame.
* Problematic if a system cannot be modularized

For more details, please refer to the following article [Software Engineering – Rapid Application Development Model (RAD)](https://www.geeksforgeeks.org/software-engineering-rapid-application-development-model-rad/).

**18.  What is the disadvantage of the spiral model?**

* Can be a costly model to use.
* Risk analysis requires highly specific expertise.
* The project’s success is highly dependent on the risk analysis phase.
* Doesn’t work well for smaller projects

For more details, please refer to the following article [Software Engineering – Spiral Model](https://www.geeksforgeeks.org/software-engineering-spiral-model/).

**19.  What is COCOMO model?**

A COCOMO model stands for Constructive Cost Model. As with all estimation models, it requires sizing information and accepts it in three forms:

* Object points
* Function points
* Lines of source code

For more details, please refer to the following article [Software Engineering – COCOMO Model](https://www.geeksforgeeks.org/software-engineering-cocomo-model/).

**20. Define an estimation of software development effort for organic software in the basic COCOMO model?**

Estimation of software development effort for organic software in the basic COCOMO model is defined as

Organic: Effort = 2.4(KLOC) 1.05 PM

**21. What is the Agile software development model?**

The agile SDLC model is a combination of iterative and incremental process models with a focus on process adaptability and customer satisfaction by rapid delivery of working software products. Agile Methods break the product into small incremental builds. Every iteration involves cross-functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing, and acceptance testing.

**Advantages:**

* Customer satisfaction by rapid, continuous delivery of useful software.
* Customers, developers, and testers constantly interact with each other.
* Close, daily cooperation between business people and developers.
* Continuous attention to technical excellence and good design.
* Regular adaptation to changing circumstances.
* Even late changes in requirements are welcomed.

For more details, please refer to the following article [Software Engineering – Agile Development Models](https://www.geeksforgeeks.org/software-engineering-agile-development-models/).

**22. Which model can be selected if the user is involved in all the phases of SDLC?**

RAD model can be selected if the user is involved in all the phases of SDLC.

**23. What are software project estimation techniques available?**

There are some software project estimation techniques available:

* PERT
* WBS
* Delphi method
* User case point

**24. What is level-0 DFD?**

The highest abstraction level is called Level 0 of DFD. It is also called context-level DFD. It portrays the entire information system as one diagram.

For more details, please refer to the following article [DFD](https://www.geeksforgeeks.org/what-is-dfddata-flow-diagram/).

**25. What is physical DFD?**

Physical DFD focuses on how the system is implemented. The next diagram to draw after creating a logical DFD is physical DFD. It explains the best method to implement the business activities of the system. Moreover, it involves the physical implementation of devices and files required for the business processes. In other words, physical DFD contains the implantation-related details such as hardware, people, and other external components required to run the business processes.

**26. What is the black hole concept in DFD?**

A block hole concept in the data flow diagram can be defined as “A processing step may have input flows but no output flows”.In a black hole, data can only store inbound flows.

**27. Mention the formula to calculate the Cyclomatic complexity of a program?**

The formula to calculate the cyclomatic complexity of a program is:

Rendered by QuickLaTeX.com

e = number of edges

n = number of vertices

p = predicates

For more details, please refer to the following article [Cyclomatic Complexity](https://www.geeksforgeeks.org/cyclomatic-complexity/).

**28. What is software re-engineering?**

It is a process of software development that is done to improve the maintainability of a software system.

**29. How to find the size of a software product?**

Estimation of the size of the software is an essential part of Software Project Management. It helps the project manager to further predict the effort and time which will be needed to build the project. Various measures are used in project size estimation. Some of these are:

* Lines of Code
* Number of entities in ER diagram
* Total number of processes in detailed data flow diagram
* Function points

**30. Mentions some software analysis & design tools?**

* Data Flow Diagrams
* Structured Charts
* Structured English
* Data Dictionary
* Hierarchical Input Process Output diagrams
* Entity Relationship Diagrams and Decision tables

**31. What is the difference between Bug and Error?**

Bug: An Error found in the development environment before the product is shipped to the customer.  
Error: Deviation for actual and the expected/theoretical value.

**32. What is the difference between Risk and Uncertainty?**

* Risk is able to be measured while uncertainty is not able to be measured.
* Risk can be calculated while uncertainty can never be counted.
* You are capable of make earlier plans in order to avoid risk. It is impossible to make prior plans for the uncertainty.
* Certain sorts of empirical observations can help to understand the risk but on the other hand, the uncertainty can never be based on empirical observations.
* After making efforts, the risk is able to be converted into certainty. On the contrary, you can’t convert uncertainty into certainty.
* After making an estimate of the risk factor, a decision can be made but as the calculation of the uncertainty is not possible, hence no decision can be made.

**33. What is a use case diagram?**

A use case diagram is a behavior diagram and visualizes the observable interactions between actors and the system under development. The diagram consists of the system, the related use cases, and actors and relates these to each other:

* **System**: What is being described?
* **Actor**: Who is using the system?
* **Use Case**: What are the actors doing?

**34. Which model is used to check software reliability?**

A Rayleigh model is used to check software reliability. The Rayleigh model is a parametric model in the sense that it is based on a specific statistical distribution. When the parameters of the statistical distribution are estimated based on the data from a software project, projections about the defect rate of the project can be made based on the model.

**35. What is CMM?**

To determine an organization’s current state of process maturity, the SEI uses an assessment that results in a five-point grading scheme. The grading scheme determines compliance with a capability maturity model (CMM) that deﬁnes key activities required at different levels of process maturity. The SEI approach provides a measure of the global effectiveness of a company’s software engineering practices and establishes ﬁve process maturity levels that are deﬁned in the following manner:

* Level 1: Initial
* Level 2: Repeatable
* Level 3: Defined
* Level 4: Managed
* Level 5: Optimizing

**36. Define adaptive maintenance?**

Adaptive maintenance defines as modifications and updations when the customers need the product to run on new platforms, on new operating systems, or when they need the product to interface with new hardware and software.

**37. In the context of modular software design, which of the combination is considered for cohesion and coupling?**

In the context of modular software design, high cohesion, and low coupling is considered.

**38. What is regression testing?**

Regression testing is defined as a type of software testing that is used to confirm that recent changes to the program or code have not adversely affected existing functionality. Regression testing is just a selection of all or part of the test cases that have been run. These test cases are rerun to ensure that the existing functions work correctly. This test is performed to ensure that new code changes do not have side effects on existing functions. Ensures that after the last code changes are completed, the above code is still valid.

**39. Black box testing always focuses on which requirement of software?**

Black box testing always focuses on the functional requirements of the software.

**40. Which of the testing is used for fault simulation?**

With increased expectations for software component quality and the complexity of components, software developers are expected to perform effective testing. In today’s scenario, mutation testing has been used as a fault injection technique to measure test adequacy. Mutation Testing adopts “fault simulation mode”.

**41. What is a function point?**

Function point metrics provide a standardized method for measuring the various functions of a software application. Function point metrics, measure functionality from the user’s point of view, that is, on the basis of what the user requests and receives in return.

**42. What is a baseline?**

A baseline is a measurement that defines the completeness of a phase. After all activities associated with a particular phase are accomplished, the phase is complete and acts as a baseline for next phase.

**43. What is the cyclomatic complexity of a module that has 17 edges and 13 nodes?**

The cyclomatic complexity of a module that has seventeen edges and thirteen nodes = E – N + 2

E = Number of edges, N = Number of nodes

Cyclometic complexity = 17 – 13 + 2 = 6

**44. A software does not wear out in the traditional sense of the term, but the software does tend to deteriorate as it evolves, why?**

The software does not wear out in the traditional sense of the term, but the software does tend to deteriorate as it evolves because  Multiple change requests introduce errors in component interactions.

**45. A cohesion is an extension of which concept?**

Cohesion refers to the degree to which Cohesion the elements inside a module belong together. is an extension of the information hiding concept.

**46. What are the three essential components of a software project plan?**

* Team structure,
* Quality assurance plans,
* Cost estimation.

**47. The testing of software against SRS is known as ….?**

The testing of software against SRS is known as acceptance testing.

**48. How to measure the complexity of software?**

To measure the complexity of software there are some methods in software engineering:

* Line of codes
* Cyclomatic complexity
* Class coupling
* Depth of inheritance

**49. Define the term WBS?**

The full form of WBS is Work Breakdown Structure. Its **Work Breakdown Structure** includes dividing a large and complex project into simpler, manageable, and independent tasks. For constructing a work breakdown structure, each node is recursively decomposed into smaller sub-activities, until at the leaf level, the activities become undividable and independent. A WBS works on a top-down approach. For more detail please refer [Work breakdown structure](https://www.geeksforgeeks.org/software-engineering-work-breakdown-structure/) article.

**50. A regression testing primarily related to which testing?**

Regression testing is primarily related to Maintenance testing.

#### **Q1. Describe the process you have for a programming task, from requirements to delivery.**

The software development process or life cycle is a structure applied to the development of a software product. There are several models for such processes (such as the agile method), each describing approaches to a variety of tasks or activities that take place during the process.

1. **Requirements analysis.** Extracting the requirements of a desired software product is the first task in creating it. While customers probably believe they know what the software is to do, it may require skill and experience in software engineering to recognize incomplete, ambiguous, or contradictory requirements.
2. **Specification.** Specification is the task of precisely describing the software to be written, in a rigorous way. In practice, most successful specifications are written to understand and fine-tune applications that were already well-developed, although safety-critical software systems are often carefully specified prior to application development. Specifications are most important for external interfaces that must remain stable.
3. **Software architecture.** The architecture of a software system refers to an abstract representation of that system. Architecture is concerned with making sure the software system will meet the requirements of the product, as well as ensuring that future requirements can be addressed.

[Read more here.](http://www.selectbs.com/analysis-and-design/what-is-a-software-development-process)

#### **Q2. What programming languages do you use? Which three do you prefer or are most familiar with?**

Interviewers expect engineers to be familiar with multiple languages. They might look for an engineer who has experience with C++ and with Java, to demonstrate the [applicant has programming chops](https://www.springboard.com/blog/software-engineering/coding-programming-interview-questions/) to rapidly pick up a new language. Python is a highly sought after language. If you are applying for a full-stack role, then you should be familiar with JavaScript frameworks like React and Node. (Here is a guide with 84 most asked [full-stack interview questions](https://www.springboard.com/blog/software-engineering/full-stack-developer-interview-questions/" \t "https://www.springboard.com/blog/software-engineering/21-software-engineering-interview-questions/_blank) and the best ways to answer them.)

Having some scripting experience with Perl or Python is also a big plus.

[Read more here.](https://hackernoon.com/5-programming-languages-every-master-developer-should-learn-a3929a8c6f69)

#### **Q3. How do you implement your error handling?**

Talk about writing tests, wrapping the code to catch exceptions, trying try/catch statements, and looking through the WOMM development process. Make sure that you have a well-thought-out answer to this question.

[Read more here.](http://www.openbookproject.net/books/mi2pwjs/ch04.html)

#### **Q4. What is the software development life cycle? What are the differences between them?**

SDLC or the Software Development Life Cycle is a process that produces software with the highest quality and lowest cost in the shortest time. SDLC includes a detailed plan for how to develop, alter, maintain, and replace a software system.

SDLC involves several distinct stages, including planning, design, building, testing, and deployment. Popular SDLC models include the[waterfall model](http://learnaccessvba.com/application_development/waterfall_method.htm),[spiral model](http://searchsoftwarequality.techtarget.com/definition/spiral-model), and[Agile model](http://istqbexamcertification.com/what-is-agile-model-advantages-disadvantages-and-when-to-use-it/).

[Read more here.](https://stackify.com/what-is-sdlc/)

#### **Q5. What has your experience been like as part of an Agile software development process, if any?**

Agile software development refers to software development methodologies centered around the idea of iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams. The ultimate value in Agile development is that it enables teams to deliver value faster, with greater quality and predictability, and greater aptitude to respond to change.

[Read more here.](https://www.cprime.com/resources/what-is-agile-what-is-scrum/)

#### **Q6. What is responsive design? What is the difference between fixed and fluid layouts?**

1. **Responsive website design.** Websites that are built with responsive design use media queries to target breakpoints that scale images, wrap text, and adjust the layout so that the website can ‘shrink to fit’ any size of screen, such as the difference between desktops and mobiles.
2. **Fluid website design.** Websites that are built with fluid design use percentages as relative indicators for widths.
3. **Fixed design.** Websites that are built using fixed design rely on fixed pixel widths. While a design with fixed dimensions can sometimes be the quickest way to get up and running, it’ll provide a less user-friendly experience across multiple devices.

[Read more here.](https://learn.onemonth.com/responsive-vs-adaptive-vs-fluid-design/)

#### **Q7. What is your process to test and find bugs in an application?**

[Software testing](https://appus.software/blog/the-importance-of-quality-assurance-management-why-you-should-consider-it) is a universally expected part of software development You need to create sets of tests and assessments to be conducted at various development stages. In fact, testing should be carried out at all stages of development, including after your main launch. Things change, platforms are updated, and errors in mobile apps that were not visible before an OS update can wreak havoc.

Usually, this means viewing the application as a whole and as their component pieces, then setting priorities in any areas that you think are more at risk than others. Tests are then conducted to confirm the functionality, and the detected defects are subsequently recorded. These defects can then be prioritized depending on their impact and severity.

Software Engineer at Dialpad

**[Read Story](https://www.springboard.com/success/pritisha-kumar)**

### **Algorithms and Data Structures Questions**

Many technical questions in software engineering interviews [quiz you on the fundamentals of algorithms and data structures](https://www.springboard.com/blog/software-engineering/data-structures-and-algorithms-interview-questions/" \t "https://www.springboard.com/blog/software-engineering/21-software-engineering-interview-questions/_blank)—in order to evaluate your baseline knowledge of these vital topics. This seems like a formal process and something that’s designed to penalize people who didn’t take a formal computer science degree since most [software engineers](https://www.springboard.com/blog/software-engineering/software-engineer-job-description/) will use libraries to abstract away efficient implementations of these data structures and algorithms. However, it’s an important part of the process.

It’s important for you to understand [how these data structures and algorithms actually work](https://www.springboard.com/blog/software-engineering/data-structures-and-algorithms/" \t "https://www.springboard.com/blog/software-engineering/21-software-engineering-interview-questions/_blank),  especially since it will come up in interview settings where you’ll have to whiteboard your solution. This means solving the problem with a paper and pen instead of a computer.  Here are a few sample questions to get you to practice.

#### **Q1. What is a stack? What are the two basic operations of a stack?**

A stack is a linear data structure with three basic operations: push (insertion of an element to the stack from the top), pop (removal of the latest element added to the stack). Some implementations of stack also allow peek, a function enabling you to see an element in a stack without modifying it. Stacks use a last-in, first-out structure – so the last element added to the stack is the first element that can be removed. Queues are a similar data structure, which work with a first-in, first-out structure. Stacks are usually implemented with an array or a linked list. You might be asked to implement a stack in an interview and to implement different operations.

[Read more here.](https://www.interviewsortout.com/stack-interview-questions/)

#### **Q2. Use Big O notation to describe QuickSort.**

A quick sort usually works best on average cases, but there are worst-case scenarios. On average, it is O(N log N), but O(N2) in the worst case. You’ll want to use quick sort in situations where average-case performance matters a lot rather than dwelling on the worst. You’ll need to have a deep and nuanced understanding of algorithms and their performance/implementation in order to answer.

[Read more here.](https://towardsdatascience.com/top-algorithms-and-data-structures-you-really-need-to-know-ab9a2a91c7b5)

#### **Q3. How does an array differ from a stack?**

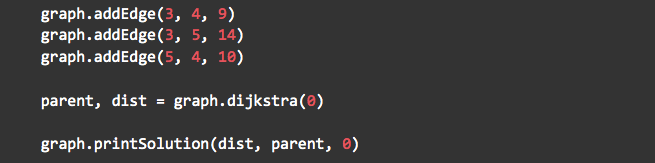
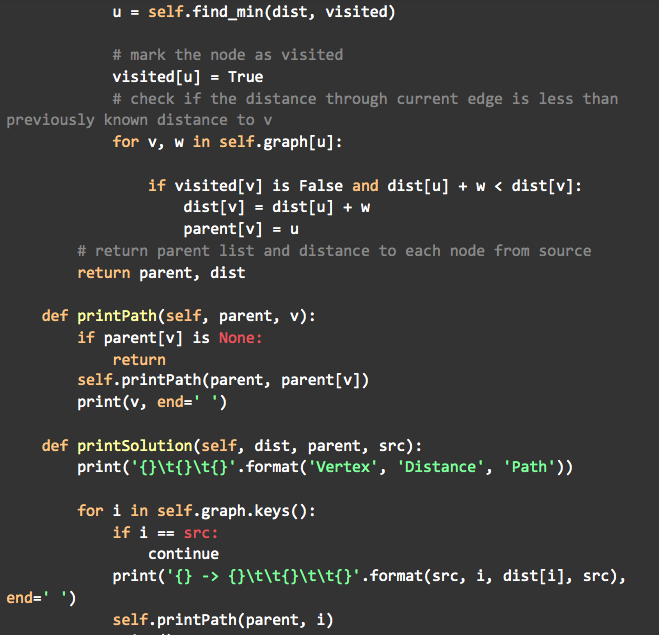
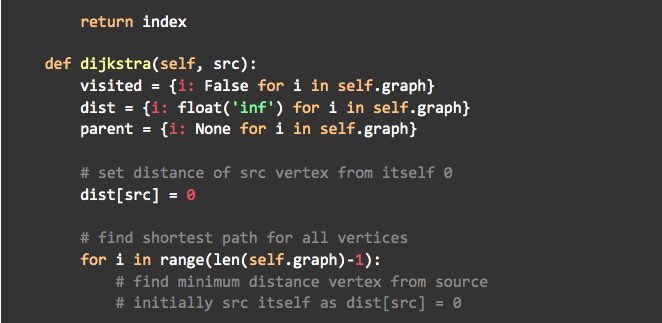
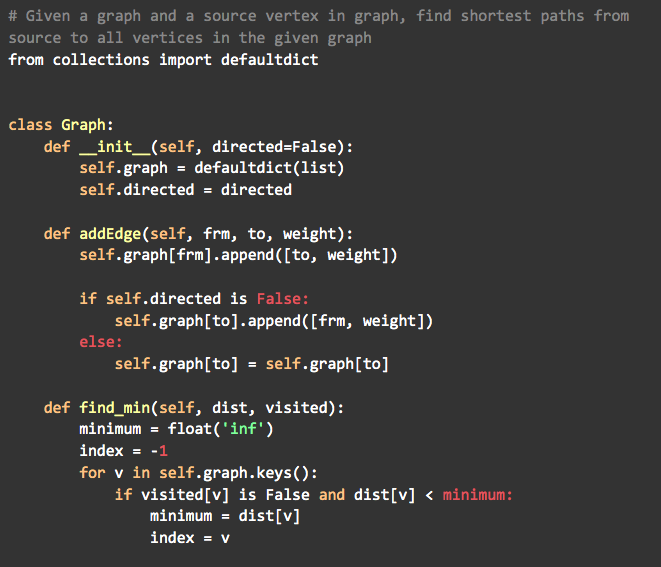
An array doesn’t have a fixed structure for how to add or retrieve data, but a stack has a strict LIFO approach (last in and first out). Questions like this will test your understanding of the nuances of data structures and the ability to memorize it.

[Read more here.](https://hackr.io/blog/data-structure-interview-questions)

#### **Q4. Implement Dijkstra’s Shortest Path in the programming language of your choice.**

Dijkstra’s algorithm is used for finding the shortest path between nodes with positive-edge weights in a graph. This is a classic algorithm question where interviewers test your understanding of how to implement an algorithm, and you’ll often see these for more senior software development roles. Dijkstra is an example: [there are others like Bellman-Ford, Floyd-Warshall](https://github.com/trekhleb/javascript-algorithms). You’ll want to study different algorithms and their implementations and practice those implementations in a variety of different manners.

[Code example here.](https://github.com/anubhavshrimal/Data-Structures-Algorithms/blob/master/Graph/DijkstraShortestPath.py)

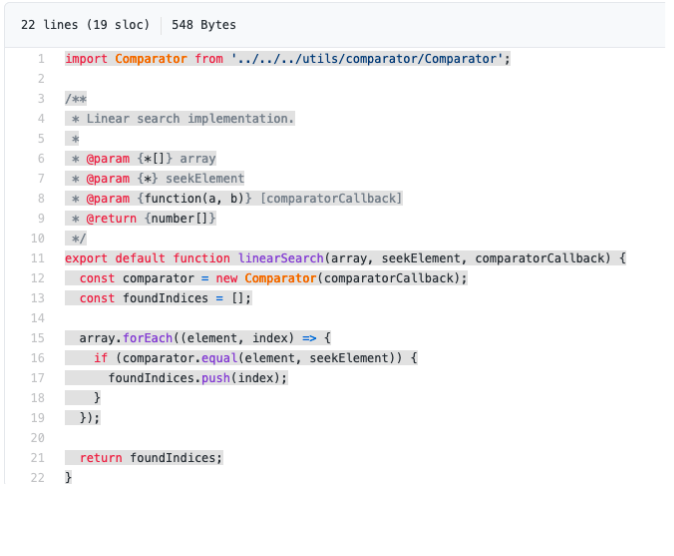


#### **Q5. Implement linear search in JavaScript.**

This will be a test of not only your algorithm and data structure knowledge but also JavaScript knowledge and implementation. You’ll want to practice in JavaScript as it’s the default language for front-end web development, and you will need to know it for front-end and full-stack positions. Showing off your ability to create algorithms in JavaScript can help demonstrate this.

Linear search is a way to find a target value within a list—it checks each element in a list and sees if it matches a certain value.

[Code example here.](https://github.com/trekhleb/javascript-algorithms/blob/master/src/algorithms/search/linear-search/linearSearch.js)



### **Quiz-style Web Developer Questions**

These questions are meant more for web development positions, especially on the freelance side, rather than harder whiteboard and algorithms questions typically seen in a software development interview. See these as more of an experiential set of questions versus the theory and algorithm-based questions listed above.

#### **Q1. What is the difference between blocking and non-blocking calls and its relationship with Node.js? Can you give an example of each?**

Blocking calls are those where the execution of additional JavaScript has to wait until a non-Javascript operation (such as something with input or output) completes or finishes. You can think of this as a synchronous action. Non-blocking calls can execute asynchronously and so therefore will have a performance advantage.

This is important because JavaScript is single-threaded, which means that it executes code in a specific order and each operation must finish executing before moving onto the next operation. JavaScript has only one call stack and one memory heap. JavaScript’s engine can help process asynchronous code on the browser.

Most of the I/O methods in Node.js offer a synchronous and asynchronous method. An example of a forced synchronous file read would be fs.readFileSync as a method, while the fs.readFile method would be asynchronous.

[Read more here.](https://nodejs.org/en/docs/guides/blocking-vs-non-blocking/)

#### **Q2. What are web workers in HTML5, and why do they matter?**

Since JavaScript is single-threaded, concurrency and simultaneous operations are difficult to execute and must be simulated with functions like setTimeout and setInterval. Web workers in HTML5 helps to run background scripts in a web application without blocking changes in the UI. In effect, web workers helps simulate multi-threading in JavaScript, allowing simultaneous scripts to run

[Read more here.](https://www.html5rocks.com/en/tutorials/workers/basics/)

#### **Q3. How do you organize CSS files? What are the pros and cons of this approach?**

This question tests your organizational ability and your familiarity with web development front-end principles, especially relevant if the role in question is more front-end focused.

Here’s an example of a file schema for CSS that would make sense:

* **reset.css:** reset and normalization styles; minimal color, border, or font-related declarations
* **typography.css:** font faces, weights, line heights, sizes, and styles for headings and body text
* **layouts.css:** styles that manage page layouts and segments, including grids
* **forms.css:** styles for form controls and labels
* **lists.css:** list-specific styles
* **tables.css:** table-specific styles
* **carousel.css:** styles required for carousel components
* **accordion.css:** styles for accordion components.”

#### **Q4. Build a single page application with multiple sections using any framework that you feel most comfortable with**

Interviewers might prefer React.js and React Router in 2020, but you can use anything you want. The purpose of this testing is to see how you build applications, even simple ones, and if you can build them at all. Oftentimes, an interviewer will observe you in a pair programming like setting, and will observe every step of your work process.

[Read more here.](https://www.kirupa.com/react/creating_single_page_app_react_using_react_router.htm)

#### **Q5. What is black box testing? What is white box testing?**

[Software Testing](https://www.geeksforgeeks.org/software-testing-basics/) can be majorly classified into two categories:

* **Black Box Testing** is a software testing method in which the internal structure/ design/ implementation of the item being tested is not known to the tester.
* **White Box Testing** is a software testing method in which the internal structure/ design/ implementation of the item being tested is known to the tester.

[Read more here.](https://www.geeksforgeeks.org/differences-between-black-box-testing-vs-white-box-testing/)

#### **Q6. What are some ways to make websites faster? Name as many different techniques as you can.**

1. Implement your own content delivery network (CDN).
2. Use adaptive images.
3. Cache, cache, cache.
4. Evaluate your plugins.
5. Combine images into CSS sprites.
6. Enable HTTP keep-alive response headers.
7. Compress your content.
8. Configure expires headers.
9. Minify JavaScript and CSS.
10. Review your hosting package.

[Read more here.](https://hostway.com/blog/10-ways-to-make-your-website-load-faster/)

#### **Q7. What is the difference between functional requirements and non-functional requirements?**

Functional requirements are the features that a developed software product is expected to perform. For example, adding a payment option at an eCommerce website will be a functional requirement. Non-functional requirements measure the usability of the application such as User Interface look and feel, Security, Performance, Interoperability, Reliability, etc.

[Read more here.](https://www.softwaretestinghelp.com/software-engineering-interview-questions/)

#### **Q8. What is the smallest building block of ReactJS?**

The smallest building blocks are React.js elements as opposed to components or props which are larger elements.

[Read more here.](https://www.skptricks.com/2018/07/react-js-quiz-react-js-interview.html)

#### **Q9. Why would you choose a microservice approach vs a monolithic app?**

If you built your app as a microservice, it’d be a combination of different services that operate independently and robustly without being dependent on one another. You might want to do this if you wanted an app with multiple points of failures or faster performance or efficiency per each app. You should be prepared to defend your decision here and to have a point of view informed by scaling issues.

[Read more here](https://sterling.com/microservices-vs-monolithic/).

### **Behavioral/Culture Fit Software Engineering Interview Questions**

#### **Q1. Tell me about a tough software development problem and how you solved it.**

Give a brief description. Make the assumption the other person doesn’t know any specialized vocabulary or industry-specific challenges. You can also ask the interviewer about their familiarity with the topic you’re about to describe and mold your answer based on the other person’s level of context (a more or less technical answer).

[Read more here.](https://www.holbertonschool.com/software-engineer-interview-questions)

#### **Q2. Do you have any personal projects? Tell me about them.**

Sometimes it’s hard to settle on an idea for a project. If you have that problem, start by making a replica of a different application with a different tech stack or something. This will get your brain pumping and eventually you’ll come up with something you’d rather do. The key isn’t coming up with a great idea. The key is to get started on something.

After you’ve worked on your replica for a while, you might notice some shortcomings in the app that you can fix. Or you might realize that you don’t want to make this replica anymore and you start on something else. The purpose of replicating an existing app isn’t to really make the replica. The purpose is to get you started on something so that you’ll find what you really want to do.

[Read more here.](https://thenextweb.com/podium/2019/08/31/personal-projects-make-you-a-better-developer/)

#### **Q3. Explain the concept of cloud computing to my older (not-very-technical) mother.**

In the simplest terms, cloud computing means storing and accessing data and programs over the Internet instead of your computer’s hard drive. Instead of storing data on your own machine, you store it on the machines of cloud service providers like Google and Amazon.

[Read more here.](https://www.pcmag.com/news/what-is-cloud-computing)

#### **Q4. Have you ever disagreed with your boss or manager? What did you do?**

Your goal is to share a story where you disagreed with your manager and you were right about the disagreement. The reason you want to be right is that your story should ideally show how competent you are at your work, which will give the hiring manager confidence in hiring you. This answer can also display other great skills such as negotiating, selling an idea, and inspiring others.

[Read more here.](https://medium.com/@Boldvue/answering-the-job-interview-question-tell-me-about-a-time-when-you-disagreed-with-your-boss-f0f95b4ccb00)

#### **Q5. Why do you want to work at [company name]? Have you used our products?**

How can you help the company succeed? Read up on what’s happening with the company and its industry. What stage of growth is the business in? Has it recently changed its product or service offerings? What competitive pressures is it facing? Consider this landscape and think, “What knowledge and experience do I have that would be especially useful to this employer in this time of growth and/or change?”

[Read more here.](https://www.roberthalf.com/blog/job-interview-tips/how-to-answer-why-do-you-want-to-work-here)

#### **Q6. When do you consider a product to be finished?**

The process of software development is a never-ending cycle. The first release of a software application is rarely “finished.” There are almost always additional features and bug fixes waiting to be designed, developed, and deployed.

Reports from [error monitoring software](https://raygun.com/platform/crash-reporting) about usability and bugs feedback into the process of software development and become new feature requests and improvements to existing features.

[Read more here.](https://raygun.com/blog/software-development-life-cycle/)

#### **Q7. Teach me about something for the next 10 minutes.**

Choose a simple topic or concept that is easy to explain and will be easy for the interviewer to understand. Making the answer fun will help to engage the interviewer. Keep the answer lighthearted. Remember, the content is not as important as the delivery and showing your communication and teaching skills.

[Read more here.](https://www.salesforcesearch.com/blog/sales-interviews-ways-answer-teach-me-something/)

#### **Q8. What web technologies are you excited about, and why do you think they’ll win/survive in the next decade?**

Choose a web technology and describe it, along with reasons (for example, technical and community support) for why it might win out against other web technologies. This question tries to gauge your passion for web development and following emerging technologies, as well as your strategic vision for the future of web development.

[Read more here.](https://designroast.org/future-of-the-web/)

#### **Q9. Do you contribute to open source projects? Have you flagged issues?**

On this question, you’ll want to flag your passion for the open-source ecosystem, as a proxy for your passion for software engineering and your ability to being proactive about contributing.

[Read more here.](https://opensource.guide/how-to-contribute/)

#### **Q10. What are your favorite resources to keep on top of software engineering?**

You’ll want to have a list of resources ready, but more importantly, you’ll want to be pretty sharp about genuinely following resources in the space. This displays your ability to learn new things and your passion for doing so, an important trait in a field that is ever-evolving. Here is one Hackernoon [resource](https://www.hackernoon.com/engineering-newsletters-that-you-should-subscribe-to-bd89dc9cd1c7) that you can learn from.

## System Design

When it comes to day-to-day software development, candidates will rely primarily on their [coding skills](https://blog.tryexponent.com/top-node-js-interview-questions). However, the best software engineers also have strong system design skills.

[System design](https://blog.tryexponent.com/how-to-nail-the-system-design-interview/) refers to developing software architecture, product design, interfaces, data, and more for a technical system according to requirements.

Expect one or two system design interview questions like those listed below. Often, companies will ask you to design a system that solves a real challenge for the company, so be sure to read your target company's engineering blog before interviewing.

This, and the structure of the system design interview, which tests a candidate's knowledge, problem-solving, and high-level thinking, makes this round a great test of how a new software engineer will perform on the job.

The best software engineering candidates have a comprehensive knowledge of distributed systems, from component-level decisions to APIs and how to scale a system reliably and efficiently.

Check out our [system design interview course](https://www.tryexponent.com/courses/system-design-interview?src=nav) to help you ace any system design question that comes your way.

1. Design a reservation and payment system for a parking garage ([Watch answer](https://www.tryexponent.com/questions/1437/design-a-reservation-and-payment-system-for-a-parking-garage)).
2. How would you design and implement a password checker given specific requirements for the password?
3. How would you create a relational schema for a calendar application? See our example answer to this question [here](https://www.tryexponent.com/questions/1205/how-would-you-design-a-relational-schema-for-a-calendar-application).
4. Design a service that supports uploading and tagging images to a travel site.
5. How long does it take to send a signal from one computer to all other computers? Watch an answer to this question [here](https://www.tryexponent.com/questions/1438/how-long-does-it-take-to-send-a-signal-from-one-computer-to-all-other-computers).
6. Design a visual landmark recognition system.
7. Design an app to be used in an amusement park. See our example answer to this question [here](https://www.tryexponent.com/questions/1586/design-app-amusement-park).
8. Design a typeahead box for a search engine. See our example answer to this question [here](https://www.tryexponent.com/questions/1601/design-typeahead-box-search-engine).
9. How does Alexa process voice commands?
10. Design Twitter.[Watch a sample answer to this question here.](https://www.tryexponent.com/courses/system-design-interview/system-design/design-twitter-video)
11. Design Facebook Messenger. Watch our co-founder, Jacob, answer this question below.

## Data Structures

The foundation of the software development process is in data structures and algorithms, hence the popularity of leetcode-esque coding questions. The best software engineering candidates understand how data structures work and practical use cases for each.

Therefore, many technical interview questions/coding exercises focus on your knowledge of data structures and appropriate algorithms (covered next).

The best way to prepare for these questions is to review the following primary components in data structures:

* [Arrays](https://www.tryexponent.com/courses/data-structures/arrays)
* [Linked Lists](https://www.tryexponent.com/courses/data-structures/linked-lists)
* [Stacks and Queues](https://www.tryexponent.com/courses/data-structures/stacks-queues)
* [Hash Tables](https://www.tryexponent.com/courses/data-structures/hash-tables)
* [Graphs and Trees](https://www.tryexponent.com/courses/data-structures/graphs-trees)
* [Heaps](https://www.tryexponent.com/courses/data-structures/heaps)

Once you've reviewed the basics, practice a few [self-guided](https://www.tryexponent.com/courses/software-engineering/swe-practice/largest-smaller-bst-key) technical interview questions to test your skills.

1. Store a list of numbers as a single number. See our example answer to this software development question [here](https://www.tryexponent.com/questions/1440/store-a-list-of-numbers-as-a-single-number).
2. How do you split up a machine learning dataset for training, evaluation, and testing?[Watch an example answer to this question here.](https://www.tryexponent.com/questions/1599/split-machine-learning-dataset-training-evaluation-testing)
3. You are given a table with varying distances from various cities. How do you find the average distance between each of the cities?
4. Determine if the given Binary Tree is a BST or not.
5. Find k-th smallest and k-th most prominent element in BST.
6. Determine if an array from 1..n has a duplicate in constant time and space.
7. If you had a vector with a lot of values in it, of which most are zeroes, how would you design a data structure to represent it?
8. Construct a Binary Tree from Ancestor Matrix
9. Check if two given binary trees are identical or not
10. Sort a Doubly Linked List using Merge Sort
11. Find number of rotations in a circularly sorted array

## Algorithms

Like data structures, engineering algorithms are another fundamental piece of the software development process. Many software engineer interview questions will be focused on them. An algorithm is a set of rules or procedures that your system or program performs during its operation.

When it comes to the software engineering interview, your hiring manager will expect you to identify and work with various algorithms that are most appropriate to whatever problem you're given.

Algorithm and data structure questions are helpful, concrete tools hiring managers use to assess a candidate's problem-solving and experience.

Be sure to review standard algorithms for searching, sorting, and notoriously-difficult dynamic programming problems. You can find many examples of these algorithms in [our SWE interview course](https://www.tryexponent.com/courses/software-engineering).

1. Write a pair of functions to serialize and deserialize a list of strings. Watch an answer to this question [here](https://www.tryexponent.com/questions/1307/write-a-pair-of-functions-to-serialize).
2. Iterate through a list of trades and return a list of buy/sell pairs and the overall profit of the transactions.
3. Merge Intervals. See our example answer to this question [here](https://www.tryexponent.com/questions/1602/merge- intervals).
4. Print left view of a binary tree. See our example answer to this question [here](https://www.tryexponent.com/questions/1473/print-left-view-of-a-binary-tree).
5. Given an nxn grid of 1 and 0s, return the number of islands in the input. See our example answer to this question [here](https://www.tryexponent.com/questions/1420/given-an-nxn-grid-of-1-and-0-s-return-the-number-of-islands-in-the-input).
6. How many hoops will a VM in one region have to go through to talk to VM in another region?
7. Find the minimum cost to reach the last cell of the matrix from its first cell.
8. Find the index of 0 to replace to get the maximum length sequence of continuous ones.
9. If you had a vector with many values in it, of which most are zeroes, how would you design a data structure to represent it?
10. Write a program to print the maximum positive sequence in a given array.

## Product Strategy

Don't worry - product strategy is probably not part of your software engineering interview. However, it is still critical for companies that their software engineers understand the product strategy that ultimately drives the software development. Having an understanding of:

* **Market Analysis:**Who your target company's users are and what they want.
* **Competitive Analysis:** What the competition looks like.
* **Product Strategy and Roadmap:** The company strategy and product pipeline may look like this.

This will give you quite an edge if you're interviewing with an ultra-competitive tech company.

If we've piqued your interest, you can check out a few [Product Strategy](https://www.tryexponent.com/courses/pm/product-strategy/intro-product-strategy) lessons in our Product Management interview course.

1. Should Facebook consolidate its messaging apps? See our example answer to this question [here](https://www.tryexponent.com/questions/1587/should-facebook-consolidate-messaging-apps).
2. Should Apple go into modular phones? See our example answer to this question [here](https://www.tryexponent.com/questions/1591/should-apple-modular-phones).
3. Why is Android strategically important to Google?
4. If you were the CEO of Microsoft, how would you increase usage for Internet Explorer?[Watch a sample answer to this question here.](https://www.tryexponent.com/questions/68/increase-internet-explorer-usage-microsoft-ceo)
5. How should Google enter the streaming space?

## ****Behavioral****

At a certain point during your software development or engineering interview, you will likely be asked some [behavioral questions](https://blog.tryexponent.com/how-to-nail-amazons-behavioral-interview-questions/)**.**

As the name suggests, [behavioral interview](https://www.tryexponent.com/courses/behavioral) questions are used by interviewers to evaluate your past [behavior and performance](https://blog.tryexponent.com/how-to-ace-google-facebook-and-amazon-behavioral-interview-questions/) in your previous positions.

Behavioral interview questions are also a great way to determine if you would mesh well with the [company culture](https://blog.tryexponent.com/facebook-google-amazon-core-values-for-your-upcoming-interview-2/).

These kinds of questions can be difficult for some. Therefore, we recommend using a framework such as STAR to ensure that you're answering your behavioral interview questions as best as possible.

Here are some of the best examples of behavioral questions you can expect:

1. Tell me about a time you had to make a decision to make short-term sacrifices for long-term gains.[Watch a sample answer to this software development behavioral question here.](https://www.tryexponent.com/questions/653/time-make-decision-short-term-sacrifices-long-term-gains)
2. Tell me about the most complex project you've worked on.[Watch a sample answer to this interview question here.](https://www.tryexponent.com/questions/926/tell-me-about-the-most-complex-project-you-ve-worked-on-)
3. If you have to work on 5 different projects, how do you prioritize?
4. How do you structure 1:1s?
5. Tell me about a situation where you had to solve a difficult problem.
6. If given a chance what would you have changed anything about a project what would it be and why ?
7. Tell me about a time where you had to lead your team through a significant change (this could be a reorganization or process). How did you handle it?
8. How should Google enter the streaming space?

## How Top Tech Companies Assess Software Engineering Candidates in 2022

Now that we've gone over the types of software engineer interview questions you'll face, we should cover how hiring managers in 2022 assess their software development candidates. Specifically, here are the primary skills or dimensions that software engineering interviews evaluate.

### Technical Skills

It should come as no surprise that the fundamental dimension of software development interviews centers on assessing a candidate's technical skills.

Hiring managers seek candidates with knowledge and experience in the [programming languages](https://blog.tryexponent.com/the-highest-paying-programming-languages), frameworks, cloud services, etc., used at the company. Not only that, hiring managers to seek candidates who can write clean code that is easy to maintain.

To evaluate the technical skills of a software engineer, hiring managers ask many previously listed technical interview questions. As a result, there's a good chance that SWE candidates will need to complete coding exercises during their interviews.

The technical skills being evaluated will also include your experience with things like the agile software development process, project management tools, and the software development life cycle more generally.

### Problem-Solving Skills

While candidates will never get an offer without the necessary technical skills, make no mistake: problem-solving is a software engineer's most important skill set.

A software developer or engineer is responsible for developing complex systems to address complex problems. Such endeavors require tremendous problem-solving skills, as you can imagine. Not only that, but high-level problem-solving skills also allow a software developer to spot or anticipate bugs while finding new ways to write faster code, no matter their programming language.

Problem-solving is the most important and the most challenging aspect of software development and being a software engineer. Therefore, one way or another, nearly all the interview questions your hiring manager will ask are meant to evaluate some component of your problem-solving abilities.



### Communication Skills

However, what might not be so unsurprising is the emphasis on the communication skills of software development candidates.

While most of your work as a software developer or engineer will undoubtedly be writing code, software developers also need to write emails, slack messages, documentation, and more while communicating with co-workers and other teams.

Your interviewer may assess these skills with questions such as "[How do you explain engineering concepts to non-technical team members](https://www.tryexponent.com/courses/people-management/practice-explaining-technical-concepts)?"

### Culture Fit

Software engineers are some of the most sought-after and highest-paid workers in today's economy. But the other side of this coin is that the competitive landscape at the top tech companies is fierce.

[FAANG companies](https://blog.tryexponent.com/how-to-ace-google-facebook-and-amazon-behavioral-interview-questions/) like Google, for instance, receive [millions](https://www.cnbc.com/2019/04/17/heres-how-many-google-job-interviews-it-takes-to-hire-a-googler.html" \l ":~:text=Year after year, Google has,chance of getting into Harvard.) of applications a year. As such, culture fit is more important than ever for hiring managers.

It's simply not enough to have the technical skills necessary for the role. In addition, candidates will need to mesh well with the corporate culture and the other employees at the company.

Generally speaking, hiring managers will ask [behavioral questions](https://www.tryexponent.com/questions?page=1&type=behavioral) such as "why are you a great match for this role?" to assess culture fit.