1. **Why testing is required?**

Ans: Software testing is required to check if the system is working as per design.

**2) What types of application we test**

Ans: **web** applications  (we open these applications in browser , ex: gmail.com)

**desktop**/windows applications (we run from our desktop, ex: notepad, word)

**Mobile** applications (we run from mobile devices, android, ios, windows phone)

**Web** **services** (SOAP/REST) (we use them as part of web applications / desktop applications/mobile applications)

**ETL** **jobs**, database validations : these jobs don’t have UI and they run in back ground to load the data (informatica jobs, ssis job etc)

      Back end/batch programs/windows services

**3) what is SDLC and different phases in SDLC?**

Ans: Software development life cycle (SDLC) is a process to develop the application.

**Different phases like:**

**Requirement Analysis and planning :** Senior team members analyze the requirements/input given by customers/business users. They will check whether the requirement is feasible or not (can be done or not). They also identify the risks associated with project.

Note: this high level requirements will be written in BRD (Business Requirement document) by Business Analyst

Define/Design : in the define stage Business Analyst define more details about requirements (which are in BRD) in the form of SRS (software requirement specification) or Use Case diagram.

As part of design,

Senior Developers write High Level Design Document (HLD)

Developers write Low Level Design Document (LLD)

Seniors Tester write Test Planning document

Implementation/Development: Developers write the code for the requirements

Testers write test cases as per SRS

Testing : Execute the test cases what we prepared in previous stage

Deployment : Release the tested code to production

Maintenance : Support team monitoring the system that is running in production

**2) what is waterfal in SDLC?**

Ans: The **waterfall** model is a sequential (non-iterative) design process, used in software development processes, in which progress is seen as flowing steadily downwards (like a **waterfall**) through the phases of conception, initiation, analysis, design, construction, testing, production/implementation and maintenance.



**3) what is the process in agile model ?**

**Agile** SDLC  **model** is a combination of iterative and incremental **process models** with focus on **process** adaptability and customer satisfaction by rapid delivery of working software product.

**Agile** Methods break the product into small incremental builds. These builds are provided in iterations.

Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In agile the tasks are divided to time boxes (small time frames) to deliver specific features for a release.

Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer.

**what is scrum methodology ?**

Ans. Scrum is an **agile** way to manage a project, usually software development. **Agile** software development with Scrum is often perceived as a methodology; but rather than viewing Scrum as methodology, think of it as a framework for managing a process.

 A product owner creates a prioritized wish list called a product backlog.

 During sprint planning, the team pulls a small chunk from the top of that wish list, a sprint backlog, and decides how to implement those pieces.

 The team has a certain amount of time — a sprint (usually two to four weeks) — to complete its work, but it meets each day to assess its progress (daily Scrum).

 Along the way, the ScrumMaster keeps the team focused on its goal.

 At the end of the sprint, the work should be potentially shippable: ready to hand to a customer, put on a store shelf, or show to a stakeholder.

 The sprint ends with a sprint review and retrospective.

 As the next sprint begins, the team chooses another chunk of the product backlog and begins working again.

**what is daily standup meeting and what we discuss ?**

A **daily stand-up meeting** is a short organizational **meeting** that is held each day. The **meeting**, generally limited to between five and fifteen minutes long, is sometimes referred to as a **stand-up**, a morning roll-call or a **daily** scrum.

The purpose of the meeting is for each team member to answer the following three questions:

1) What did you do yesterday?

2) What will you do today?

3) Are there any impediments in your way?

**what is user story/feature/sprint back log items and tasks in user story?**

A user story - simply put, is a way to define a software feature from an end-user perspective.

A task is a piece of work that needs doing, usually in order to build toward a bigger story. As such, it does not have independent deliverable functionality or generate business value, and, unlike a story, it normally is not a vertical (end-to-end) slice. Most tasks tend to be for programmers, but they could be for anyone on the team.

**what is sprint planning and spring retro ?**

In [Scrum](http://www.open.collab.net/nonav/community/swp/training/IntroToScrum/Intro_to_scrum.htm" \o "Introduction to Scrum" \t "_blank), every iteration begins with a [sprint planning meeting](http://www.open.collab.net/nonav/community/swp/training/SprintPlanningMeeting/SprintPlanningMeeting.htm" \o "example Scrum Sprint Planning Meeting video" \t "_blank). At this meeting, the [Product Owner](http://scrumreferencecard.com/scrum-reference-card/#Scrum-Roles) and the team negotiate which stories a team will tackle that sprint. This meeting is a time-boxed *conversation* between the Product Owner and the team.

After the sprint review meeting, the team and the Scrum Master get together in private for the retrospective meeting. During this meeting, the team inspects and adapts their process. When the Scrum Master and outer organization create an environment of psychological safety, team members can speak frankly about what occurred during the Sprint and how they felt about it.

**what is burndown chart and velocity ?**

Its purpose is to enable that the project is on the track to deliver the expected solution within the desired schedule. Simple **Burndown Chart**. The rate of progress of a Scrum Team is called "**velocity**". It expresses the amount of e.g. story points completed per iteration.

**what is product backlog item and sprint backlog items ?**

Product Backlog Items (PBIs) are the elements that make up the Product Backlog.

The sprint backlog is a list of tasks identified by the Scrum team to be completed during the scrum print. During the sprint planning meeting, the team selects some number of product backlog items, usually in the form of user stories, and identifies the tasks necessary to complete each user story. Most teams also estimate how many hours each task will take someone on the team to complete.

**what is user acceptance criteria test cases** ?

User acceptance testing (UAT) is the last phase of the software testing process. During UAT, actual software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications. UAT is one of the final and critical software project procedures that must occur before newly developed software is rolled out to the market.

**what is v model?**

The **V** - **model** is SDLC **model** where execution of processes happens in a sequential manner in **V**-shape. It is also known as Verification and Validation **model**. **V** - **Model** is an extension of the waterfall **model** and is based on association of a testing phase for each corresponding development stage.

Under V-Model, the corresponding testing phase of the development phase is planned in parallel. So there are Verification phases on one side of the .V. and Validation phases on the other side. Coding phase joins the two sides of the V-Model.

The below figure illustrates the different phases in V-Model of SDLC.



**what is STLC?**

Software Testing Life Cycle (**STLC**) is defined as a sequence of activities conducted to perform Software Testing. It consists of series of activities carried out methodologically to help certify your software product.

Diagram - Different stages in Software Test Life Cycle

[](http://cdn.guru99.com/images/stories/software-test-life-cycle.jpg)

**what is defect?**

A programmer while designing and building the **software** can make mistakes or error. These mistakes or errors mean that there are flaws in the **software**. These are called **defects**. When actual result deviates from the expected result while **testing** a **software** application or product then it results into a **defect**.

* When actual result deviates from the expected result while testing a software application or product then it results into a defect. Hence, any deviation from the specification mentioned in the product functional specification document is a defect. In different organizations it’s called differently like bug, issue, incidents or problem.
* When the result of the software application or product does not meet with the end user expectations or the software requirements then it results into a Bug or Defect. These defects or bugs occur because of an error in logic or in coding which results into failure or unpredicted or unanticipated results.

**how to arise a defect and what we specify while logging defect?**

**defect lifecycle ?**

**Defect life cycle** is a cycle which a defect goes through during its lifetime. It starts when defect is found and ends when a defect is closed, after ensuring it’s not reproduced.

**life cycle includes following steps or status:**

1. **New:**  When a defect is logged and posted for the first time. It’s state is given as new.
2. **Assigned:**  After the tester has posted the bug, the lead of the tester approves that the bug is genuine and he assigns the bug to corresponding developer and the developer team. It’s state given as assigned.
3. **Open:** At  this state the developer has started analyzing and working on the defect fix.
4. **Fixed:** When developer makes necessary code changes and verifies the changes then he/she can make bug status as ‘Fixed’ and the bug is passed to testing team.
5. **Pending retest:**  After fixing the defect the developer has given that particular code for retesting to the tester. Here the testing is pending on the testers end. Hence its status is pending retest.
6. [**Retest**](http://istqbexamcertification.com/what-is-retesting/)**:**  At this stage the tester do the retesting of the changed code which developer has given to him to check whether the defect got fixed or not.
7. [**Verified**](http://istqbexamcertification.com/what-is-verification-in-software-testing-or-what-is-software-verification/)**:** The tester tests the bug again after it got fixed by the developer. If the bug is not present in the software, he approves that the bug is fixed and changes the status to “verified”.
8. **Reopen:** If the bug still exists even after the bug is fixed by the developer, the tester changes the status to “reopened”. The bug goes through the life cycle once again.
9. **Closed:** Once the bug is fixed, it is tested by the tester. If the tester feels that the bug no longer exists in the software, he changes the status of the bug to “closed”. This state means that the bug is fixed, tested and approved.
10. **Duplicate:** If the bug is repeated twice or the two bugs mention the same concept of the bug, then one bug status is changed to “duplicate**“.**
11. **Rejected:** If the developer feels that the bug is not genuine, he rejects the bug. Then the state of the bug is changed to “rejected”.
12. **Deferred:** The bug, changed to deferred state means the bug is expected to be fixed in next releases. The reasons for changing the bug to this state have many factors. Some of them are [**priority**](http://istqbexamcertification.com/what-is-the-difference-between-severity-and-priority/)of the bug may be low, lack of time for the release or the bug may not have major effect on the software.
13. **Not a bug:**  The state given as “Not a bug” if there is no change in the functionality of the application. For an example: If customer asks for some change in the look and field of the application like change of colour of some text then it is not a bug but just some change in the looks of the  application.

**Different types of testing:**

Different types of tests (GUI testing, Functional testing, **Regression testing**, Smoke testing, load testing, stress testing, security testing, stress testing, ad-hoc testing etc.,) are carried out to complete **system testing**.

**What is unit testing?**

**Unit testing** is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. **Unit testing** can be done manually but is often automated.

Unit testing is a component of a pragmatic methodology that takes a meticulous approach to building a product by means of continual testing and revision. Test-driven development requires that developers first write failing unit tests. Then they write code and refactor the application until the test passes. TDD typically results in an explicit and predictable code base.

Unit testing involves only those characteristics that are vital to the performance of the unit under test. This encourages developers to modify the [source code](http://searchsoa.techtarget.com/definition/source-code) without immediate concerns about how such changes might affect the functioning of other units or the program as a whole.

**when do we use regression testing?**

**Regression testing** is the process of **testing** changes to computer programs to make sure that the older programming still works with the new changes. **Regression testing** is a normal part of the program development process and, in larger companies, is **done** by code **testing** specialists.

Regression testing ensures that little changes don't break software. Good regression testers need to know what they're looking for, and this guide explains how.

**What is integration testing?**

Ans: **Integration** is the phase in software **testing** in which individual software modules are combined and **tested** as a group. It occurs after unit **testing** and before validation **testing**.

**when do we use integration testing?**

The purpose of integration testing is to verify functional, performance, and reliability [requirements](https://en.wikipedia.org/wiki/Requirement) placed on major design items. These "design items", i.e., assemblages (or groups of units), are exercised through their interfaces using [black-box testing](https://en.wikipedia.org/wiki/Black-box_testing), success and error cases being simulated via appropriate parameter and data inputs.

**when do we use smoke testing and sanity testing?**

Smoke Testing is a kind of Software Testing performed after software build to ascertain that the critical functionalities of the program is working fine. It is executed "before" any detailed functional or regression tests are executed on the software build. The purpose is to reject a badly broken application, so that the QA team does not waste time installing and testing the software application.

Sanity testing is a kind of Software Testing performed after receiving a software build, with minor changes in code, or functionality, to ascertain that the bugs have been fixed and no further issues are introduced due to these changes. The goal is to determine that the proposed functionality works roughly as expected. If sanity test fails, the build is rejected to save the time and costs involved in a more rigorous testing.

**what is alpha and beta testing?**

**Alpha testing** is a type of acceptance **testing**; performed to identify all possible issues/bugs before releasing the product to everyday users or public.  **Alpha testing** is carried out in a lab environment and usually the testers are internal employees of the organization.

Beta Testing of a product is performed by "real users" of the software application in a "real environment" and can be considered as a form of external user acceptance testing.

 Beta version of the software is released to a limited number of end-users of the product to obtain feedback on the product quality. Beta testing reduces product failure risks and provides increased quality of the product through customer validation.

**when do we use white box testing and block box testing?**

**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.

**what we will do if come across any critical severity issue before release day?**

The fact that the defect has been found close to the deadline is, in the short term, irrelevant. Your team has found a high severity defect, so you report it. Given the short timescales, you ensure that everyone who needs to know about it knows about it, so they have the information they need to determine -their- best course of action as soon as possible.

You must -absolutely- not ever hold off from reporting an issue, at least to your local management structure. That would, at the very least, ruin the reputation of your team and could potentially have much more serious consequences.

**when do we use automation testing?**

Test engineers strive to catch them before the product is released but they always creep in and they often reappear, even with the best manual **testing processes**. Test Automation software is the best way to increase the effectiveness, efficiency and coverage of your software testing.

**what tester will do in each phase of SDLC?**

**In** phases of **SDLC**:

* Requirements gathering and Analysis

In this phase of SDLC, suitable necessities of system are accumulated. All adjacent methods should be in focus. All types of estimation and examination of user needs are done in this phase.

* System Design

In the second phase a basic system planning is done. After collecting the all statistics and data, a system design is done.

* Implementation

In the next phase implementation of project is done. Respect to the system design, correct development is made to expand that design. According to the project programming language will be chosen.

* System Testing

After the implementation phase, system testing phase take place to recognize the result of application. Testing is done to recognize the original result and the predictable result.

* Operation Maintenance

It is the ultimate phase of SDLC, where the application which is implemented is spread to users who are answerable for conserving and using it for appropriate actions. The implemented application should be available for any adjustment to do in coding.

**difference between load and performance testing?**

Performance testing is the testing, which is performed, to ascertain how the components of a system are performing, given a particular situation. Resource usage, scalability and reliability of the product are also validated under this testing. This testing is the subset of performance engineering, which is focused on addressing performance issues in the design and architecture of software product.

Load testing is meant to test the system by constantly and steadily increasing the load on the system till the time it reaches the threshold limit. It is the simplest form of testing which employs the use of automation tools such as LoadRunner or any other good tools, which are available. Load testing is also famous by the names like **volume testing** and **endurance testing**.

**different types of non-functional testing types?**

* Load/Performance testing.
* Compatibility testing.
* Localization testing.
* Security testing.
* Reliability testing.
* Stress testing.
* Usability testing.
* Compliance testing.

**what is test case?**

A **test case**, in software engineering, is a set of conditions under which a tester will determine whether an application, software system or one of its features is working as it was originally established for it to do.

what is test planning/test strategy document

Ans: Test plan document contains different section like

       Types of testing :

       Exit and Entry criteria :

**what is TDD and BDD (cucumber framework)**

**Cucumber** is a testing **framework** which supports Behavior Driven Development (**BDD**). It lets us define application behavior in plain meaningful English text using a simple grammar defined by a language called Gherkin.

ts also called test-driven design, is a method of software development in which unit testing is repeatedly done on source code. Write your tests watch it fails and then refactor it.**The concept is we write these tests to check if the code we wrote works fine.** After each test, refactoring is done and then the same or a similar test is performed again. The process is iterated as many times as necessary until each unit is functionally working as expected. **TDD was introduced first by XP**. I believe I have explained enough in simple terms.

Behavior-driven development combines the general techniques and principles of TDD with ideas from domain-driven design.

**what is priority and severity in defect?**

**Severity** is defined as the degree of impact a **defect** has on the development or operation of a component application being tested

Priority is defined as the order in which a defect should be fixed. Higher the priority the sooner the defect should be resolved.

Defects that leave the software system unusable are given higher priority over defects that cause a small functionality of the software to fail.

**how to estimate test cases?**

1. 3-Point Software Testing Estimation Technique.
2. Use – Case Point Method:
3. Work Breakdown Structure.
4. Wideband Delphi technique.
5. Function Point/Testing Point Analysis.
6. Percentage of development effort method.
7. Percentage distribution.
8. Best Guess.

**what is most challenge defect u came across?**

**how to deal the production defects?**

Ans: normally end user will report this issue.

       we need to talk to them (end users) and reproduce the issue with in staging environment

  Create defect in defect tool under the production release version

 developers will fix the issuewe (QA) test the issue on production version code (stageing) and release the fix to proudction after we verify

 we have to create a defect on current **sprint/release** so that developer will add this code to the current sprint/release

**If we dont have time to test call test cases what we will do ?**

**how we learn the functionality of system?**

**what are the tools to manage defects/stories?**

* Stryka: Details: Stryka is a cutting-edge enterprise test management tool, built from the ground up using the latest web and mobile technologies.
* Bugzilla:
* Lean Testing.
* JIRA:
* Mantis:
* Trac:
* Redmine:

**who will assign the work?**

**QA lead**

**types of test metrics we use normally ?**

* **Process Metrics:** It can be used to improve the process efficiency of the SDLC ( Software Development Life Cycle)
* **Product Metrics:** It deals with the quality of the software product
* **Project Metrics:** It can be used to measure the efficiency of a project team or any tools being used by the team members

Manual test metrics is classified into two classes

* **Base Metrics**
* **Calculated Metrics**
* Base metrics is the raw data collected by Test Analyst during the test case development and execution (**# of test cases executed, # of test cases**). While, calculated metrics is derived from the data gathered in base metrics. Calculated metrics is usually tracked by the test manager for test reporting purpose (**% Complete, % Test Coverage**).

**what is traceability matrix?**

A **traceability matrix** is a document, usually in the form of a table, used to assist in determining the completeness of a relationship by correlating any two baselined documents using a many-to-many relationship comparison.

Common usage is to take the identifier for each of the items of one document and place them in the left column. The identifiers for the other document are placed across the top row. When an item in the left column is related to an item across the top, a mark is placed in the intersecting cell. The number of relationships are added up for each row and each column. This value indicates the mapping of the two items. Zero values indicate that no relationship exists. It must be determined if a relationship must be made. Large values imply that the relationship is too complex and should be simplified.

**what are typical environments we have in projects**

**What is staging environment**

A stage or **staging environment** is an **environment** for testing that exactly resembles the production **environment**. In other words, it's a complete but independent copy of the production **environment**, including the database. **Staging** provides a true basis for QA testing because it precisely reproduces what is in production.

A staging environment is meant to have everything as closely replicated to the production environment as possible so that you can maximize your chances of finding any bugs before you release the software in production. Even the hardware that is used for the staging environment is often the same as the hardware used in the production environment – this is a good practice when possible.

**what is development environment**

In computer program and software product **development**, the **development environment** is the set of processes and programming tools used to create the program or software product. The term may sometimes also imply the physical **environment**.

Development Server: Here is where the developer tests code and checks whether the application runs successfully with that code. Once the application has been tested and the developer feels that the code is working fine, the application then moves to the staging server.

**what is QA environment and what is production environment ?**

A **QA environment** is where you test your upgrade procedure against data, hardware, and software that closely simulate the Production **environment** and where you allow intended users to test the resulting Waveset application. A

Production **environment** is where the Waveset application is actually available for business use.

Production environment is a term used mostly by developers to describe the setting where software and other products are actually put into operation for their intended uses by end users. A production environment can be thought of as a real-time setting where programs are run and hardware setups are installed and relied on for organization or commercial daily operations.

One way to define a production environment is by contrasting it with a testing environment. In a testing environment, a product is still being used theoretically. Users, typically engineers, look for bugs or design flaws. In the production environment, the product has been delivered and needs to work flawlessly.