**1).Why testing is required?**  
1. Testing is always needed for correctly understand the fault errors in software during its development phases.  
2. It is necessary because it always ensures the users or customers satisfaction and reliability of the application.  
3. It is needed in software development to increase the reliability and quality of the software.  
4. **Testing is needed** to provide the various facilities to the users like delivery of high quality software or application, lower maintenance costs and more accurate and reliable results also.  
5. Testing is necessary because of effective optimum performance of system and capacity utilization, software reliability, quality, and system or application assurance.  
6. It is necessary because it is including in the project plan, and to stay in business.  
7. It is needed to prove that software or application has no faults, because failures can be very expensive in nature.  
8. Software Testing is needed to learn more about the reliability of the software and application.

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

**Accessibility Testing**  
When doing accessibility testing, the aim of the testing is to determine if the contents of the website can be easily accessed by disable people. Various checks such as color and contrast (for color blind people), font size for visually impaired, clear and concise text that is easy to read and understand.

**API Testing**  
API testing is a type of testing that is similar to unit testing. Each of the Software APIs are tested as per API specification. API testing is mostly done by testing team unless APIs to be tested or complex and needs extensive coding. API testing requires understanding both API functionality and possessing good coding skills.

**Automated testing**  
This is a testing approach that makes use of testing tools and/or programming to run the test cases using software or custom developed test utilities. Most of the automated tools provided capture and playback facility, however, there are tools that require writing extensive scripting or programming to automate test cases.  
**Beta Testing**  
This is a formal type of software testing that is carried out by end customers before releasing or handing over software to end users. Successful completion of Beta testing means customer acceptance of the software.

**Black Box testing**  
Black box testing is a software testing method where testers are not required to know coding internal structure of the software. Black box testing method relies on testing software with various inputs and validating results against expected output.

**Backward Compatibility Testing**  
Type of software testing performed to check that the newer version of the software can work successfully on top of the previous version of the software and that the newer version of the software works as fine with table structure, data structures and files that were created by the previous version of the software.

**Big Bang Integration testing**  
This is one of the integration testing approaches, in Big Bang integration testing all or all most all of the modules are developed and then coupled together.

**Bottom up Integration testing**  
Bottom-up integration testing is an integration testing approach where testing starts with smaller pieces or sub-systems of the software till all the way up covering entire software system. Bottom-up integration testing begins with small portions of the software and eventually scale up in terms of size, complexity, and completeness.

**Branch Testing**  
Is a white box testing method for designing test cases to test code for every branching condition. Branch testing method is applied during unit testing.

**Component Testing**  
This type of software testing is performed by developers. Component testing is carried out after completing unit testing. Component testing involves testing a group of units as code together as a whole rather than testing individual functions, methods.

**Dynamic Testing**  
Testing can be performed as Static Testing and Dynamic testing, Dynamic testing is a testing approach where testing can be done only by executing code or software are classified as Dynamic Testing. Unit testing, Functional testing, regression testing, performance testing etc.,

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

**SDLC** is a **process** followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development **process**.

1. Requirement gathering and analysis
2. Design
3. Implementation or coding
4. Testing
5. Deployment
6. Maintenance

**4). what is waterfall method?**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.

**5).what is agile method?**Not a methodology! The Agile movement seeks alternatives to traditional project management. Agile approaches help teams respond to unpredictability through incremental, iterative work cadences and empirical feedback. Agilists propose alternatives to waterfall, or traditional sequential development.

**6). what is scrum methodology**  
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.

**7).What is the process in agile methodology?**

**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.

**8)**. **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.

**9).What is product backlog items?**

The agile product backlog in [Scrum](https://www.mountaingoatsoftware.com/agile/scrum) is a prioritized features list, containing short descriptions of all functionality desired in the product. When applying Scrum, it's not necessary to start a project with a lengthy, upfront effort to document all requirements. Typically, a Scrum team and its product owner begin by writing down everything they can think of for agile backlog prioritization. This agile product backlog is almost always more than enough for a first sprint. The Scrum product backlog is then allowed to grow and change as more is learned about the product and its customers.

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

Though they sound similar in a functional manner, user stories and tasks are quite different aspects of agile methodology. Still, many of us use the terms user story and tasks interchangeably. Not only this causes confusion, but also keeps you from reaping the full benefits of your agile work culture. Until and unless you clearly know the terms and their meanings, you will not be able to follow the best practices. So, let’s try to clearly understand the difference between user stories and tasks.

 the difference that user stories go to the product backlog and tasks go to sprint backlog after sprint planning is not valid. It doesn’t explain the difference but just tell what goes where. We are looking to understand the exact meaning and the proper application of the terms.

11) **what is sprint planing meeting?**

**Sprint planning** is a collaborative effort involving a ScrumMaster, who facilitates the **meeting**, a Product Owner, who clarifies the details of the product backlog items and their respective acceptance criteria, and the Entire Agile Team, who define the work and effort necessary to meet their **sprint** commitment

**12) what is sprint review meeting?**

In Scrum, each **sprint** is required to deliver a potentially shippable product increment. This means that at the end of each **sprint**, the team has produced a coded, tested and usable piece of software. So at the end of each **sprint**, a **sprint review meeting**is held

**13) what is sprint retrospective?**

The **sprint retrospective** is a meeting facilitated by the ScrumMaster at which the team discusses the just-concluded **sprint** and determines what could be changed that might make the next **sprint** more productive.

**14) what is sprint grooming?**

While backlog refinement (also called grooming) was not originally a formal meeting in the Scrum framework, Ken Schwaber, who founded Scrum, advises teams to dedicate five percent of every sprint to this activity. As with Scrum’s other meetings, the grooming should take place at the same time and place and for the same duration each sprint.

**15) 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.

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

Firstly, the **criteria** by which the software is considered to be “working” needs to be assembled. These are likely to be collated from the system requirements, and **user**stories. Next, a set of **UAT test cases** must be created. Centric defines a **UAT test case** as: ... Each **case** covers a specific usage scenario of the software.

**17) 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.

**18) what is STLC?**

**Software Testing Life Cycle** (**STLC**) is the testing process which is executed in systematic and planned manner. In **STLC** process, different activities are carried out to improve the quality of the product. And Test Execution.

**19) what is defect?**

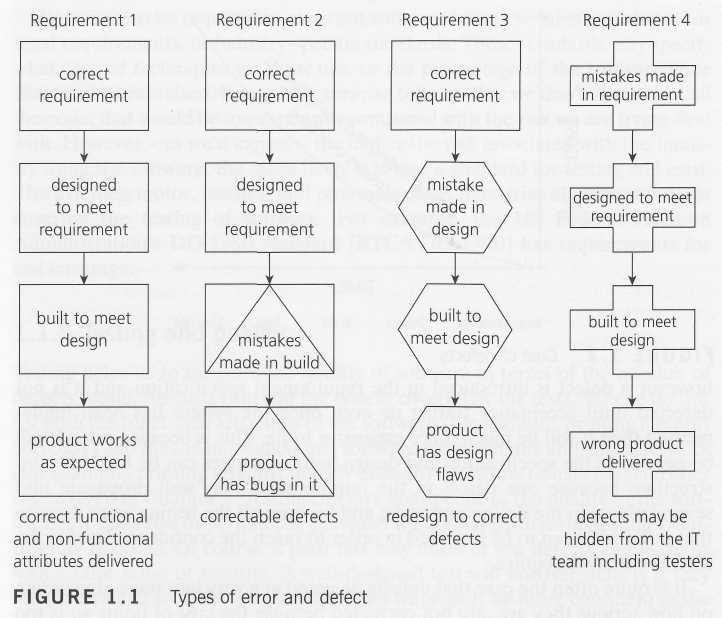
A **defect is an error or a bug**, in the application which is created. A programmer while designing and building the software can make mistakes or error. These mistakes or errors mean that there are flaws in the software. These are called defects.

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

We can see that **Requirement 1** is implemented correctly – we understood the customer’s requirement, designed correctly to meet that requirement, built correctly to meet the design, and so deliver that requirement with the right attributes: [**functionally**](http://istqbexamcertification.com/what-is-functional-testing-testing-of-functions-in-software/), it does what it is supposed to do and it also has the right [**non-functional**](http://istqbexamcertification.com/what-is-non-functional-testing-testing-of-software-product-characteristics/) attributes, so it is fast enough, easy to understand and so on.

**21) defect lifecycle?**

We can see that **Requirement 1** is implemented correctly – we understood the customer’s requirement, designed correctly to meet that requirement, built correctly to meet the design, and so deliver that requirement with the right attributes: [**functionally**](http://istqbexamcertification.com/what-is-functional-testing-testing-of-functions-in-software/), it does what it is supposed to do and it also has the right [**non-functional**](http://istqbexamcertification.com/what-is-non-functional-testing-testing-of-software-product-characteristics/) attributes, so it is fast enough, easy to understand and so on.

With the other requirements, errors have been made at different stages. **Requirement 2** is fine until the software is coded, when we make some mistakes and introduce defects. Probably, these are easily spotted and corrected during testing, because we can see the product does not meet its design specification.

The defects introduced in **Requirement 3** are harder to deal with; we built exactly what we were told to but unfortunately the designer made some mistakes so there are defects in the design. Unless we check against the requirements definition, we will not spot those defects during testing. When we do notice them they will be hard to fix because design changes will be required.

The defects in **Requirement 4** were introduced during the definition of the requirements; the   product has been designed and built to meet that flawed requirements definition. If we test the product meets its requirements and design, it will pass its tests but may be rejected by the user or customer. Defects reported by the customer in acceptance test or live use can be very costly. Unfortunately, requirements and design defects are not rare; assessments of thousands of projects have shown that defects introduced during requirements and design make up close to half of the total number of defects.

22). 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.

**23). 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.

**24) What is integration testing?**

**ntegration testing** (sometimes called **integration** and **testing**, abbreviated I&T) 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**.

**25) when do we use integration testing?**

There are two major ways of carrying out an integration test, called the bottom-up method and the top-down method. Bottom-up integration testing begins with [unit testing](http://searchsoftwarequality.techtarget.com/definition/unit-testing), followed by tests of of progressively higher-level combinations of units called modules or builds. In top-down integration testing, the highest-level modules are tested first and progressively lower-level modules are tested after that. In a comprehensive software development environment, bottom-up testing is usually done first, followed by top-down testing. The process concludes with multiple tests of the complete application, preferably in scenarios designed to mimic those it will encounter in customers' computers, systems and [network](http://searchnetworking.techtarget.com/definition/network)s.

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

* Smoke testing originated in the hardware testing practice of turning on a new piece of hardware for the first time and considering it a success if it does not catch fire and smoke. In software industry, smoke testing is a shallow and wide approach whereby all areas of the application without getting into too deep, is tested.
* A smoke test is scripted, either using a written set of tests or an automated test
* A Smoke test is designed to touch every part of the application in a cursory way. It’s shallow and wide.
* Smoke testing is conducted to ensure whether the most crucial functions of a program are working, but not bothering with finer details. (Such as build verification).
* Smoke testing is normal health check up to a build of an application before taking it to testing in depth.

**27)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.

**28)What is UAT?**

**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.

**29) 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

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

Black-box testing (also known as functional testing) treats software under test as a black-box without knowing its internals. Tests are using software interfaces and trying to ensure that they work as expected. As long as functionality of interfaces remains unchanged, tests should pass even if internals are changed. Tester is aware of what the program should do but does not have the knowledge of how it does it. Black-box testing is most commonly used type of testing in traditional organizations that have testers as a separate department, especially when they are not proficient in coding and have difficulties to understand the code. It provides **external perspective** of the software under test.

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) looks inside the software that is being tested and uses that knowledge as part of the testing process. If, for example, exception is thrown under certain conditions, test might want to reproduce those conditions. White-box testing requires internal knowledge of the system and programming skills. It provides **internal perspective** of the software under test.

**31) what we will do if we don't have a time to test all stories/ execute test cases?**

If you started with an inaccurate expectation, things are bound to fail. A good test estimate must take the following into account:

* **Time for preparatory tasks –** We are talking about tasks such as:
  + Identifying and putting together a regression suite
  + Creating Test data
  + Time to determine test readiness (E.g.: Smoke/Sanity Test), etc.
* **Test case maintenance**: Test cases are long-term usage assets. They are sure to undergo minor updates during execution. It is recommended that for new products up to 30% of your test execution time should be allocated for these minor maintenance tasks. All teams and projects might not need 30%, but do allocate some time and effort for this task.
* [**Ad-hoc**](http://www.softwaretestinghelp.com/ad-hoc-testing/)**/Exploratory testing**– The count of scripted tests is a major denominator for test estimation numbers. However, no test team in this world will deny exploring your software even if the model is dominantly scripted.
* **Reporting/Communication**– This includes triage/stand up meetings, updating work management tools etc.
* **Contingency factor:** Standards recommend 25-30% buffer to your original estimates. But teams can rarely afford it. Even then, leave a little breathing room, when possible.
* **Team and its capabilities:** If you have a new team or if they are using a tool for the first time, you might need to set some time aside for training. Tailor your estimates based on your team you are working with.

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

This is a simple, if uncomfortable, situation to be in. Unfortunately, it does happen from time to time and you need to be ready for it.

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.

**33) 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.

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

n the SDLC the tester should be involved at the beginning of the discussion, inception, analysis. Most will try and say not unit the Software Requirements Spec. (SRS) is written ... but not true. Testing should be involved from inception to the production phase of a project.

**35) difference between load and performance testing?**

A web developer called me the other day and asked, “What’s the difference between load testing vs performance testing?”

I pondered for a moment and replied, “Not really.” Hmmmm…now I needed to support my answer. Thus began a hunt for good answers to my client’s question.

Searching Google, gleaning from our interviews, and asking respected colleagues were parts of my pursuit for the truth about load vs. perf. Sorry to abbreviate, but I’ve heard cool engineers shorten it to “perf”, and I desperately want to be cool. (in a geeky way)

The answer isn’t black and white. However, I have heard software testing professionals say load testing is more aligned with black box testing and performance testing with white box testing. Thus, performance involves getting inside the various components of a system.

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

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

**37) what is test case?**

A **test case**, in [software engineering](https://en.wikipedia.org/wiki/Software_engineering), is a set of conditions under which a tester will determine whether an [application](https://en.wikipedia.org/wiki/Software_application), [software system](https://en.wikipedia.org/wiki/Software_system) or one of its features is working as it was originally established for it to do. The mechanism for determining whether a software program or system has passed or failed such a test is known as a [*test oracle*](https://en.wikipedia.org/wiki/Oracle_(software_testing)). In some settings, an oracle could be a [requirement](https://en.wikipedia.org/wiki/Requirement) or [use case](https://en.wikipedia.org/wiki/Use_case), while in others it could be a [heuristic](https://en.wikipedia.org/wiki/Heuristic). It may take many test cases to determine that a software program or system is considered sufficiently scrutinized to be released. Test cases are often referred to as [*test scripts*](https://en.wikipedia.org/wiki/Test_script), particularly when written - when they are usually collected into [test suites](https://en.wikipedia.org/wiki/Test_suite).

**37) what is test planning/test strategy document**

A **Test Plan Documents** the **strategy** that will be used to verify and ensure that a product or system meets its design specifications and other requirements. ... The**Test Strategy document** describes the scope, approach, resources and schedule for the **testing** activities of the project.

**38) what is Exit and Entry criteria :**

Criteria comes in every phase of testing( Static & Dynamic testing) They are 4 types of criteria. 1. Enter Criteria 2. Suspension Criteria 3. Resuspension Criteria & 4. Exit Criteria 1. Enter criteria: This is the first stage and starting of the Phases2. Suspension Criteria: when your sending the document to verify by developer& Test lead.at that time Suspension Criteria comes screen3. Resuspension Criteria: there is any changes want to done in your document . at that time Resuspension Criteria comes on Screen.4. Exit Criteria: You completed your testing .at that time Exit Criteria comes on Screen

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

**Behavior-driven Development** (**BDD**) is an agile software development practice that enhances the paradigm of **Test Driven Development** (**TDD**) and acceptance tests, and encourages the collaboration between developers, quality assurance, domain experts, and stakeholders.

**40) how do we write test cases in BDD format**

Traditionally, structured test, especially long scenarios are commonly written in the step, expected, actual format. Behaviour Driven Design replaces this with the style of "Given", "When" "Then", for example

Given a working ATM And invalid credit card

When the card is inserted

And a cash withdrawl is requested

Then no cash should be provided

And the card should be retained

I want to start use BDD on complex scenarios on a project, and the question that I have is: Does this new style of test case work for larger, real world, end-to-end scenarios or is it more appropriate for simpler "atomic" or unit test style test cases?

**41) 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 defines the order in which we should resolve a defect. Should   we fix it now, or can it wait? This priority status is set by the tester to the developer mentioning the time frame to fix the defect. If high priority is mentioned then the developer has to fix it at the earliest. The priority status is set based on the customer requirements.**For example:**If the company name is misspelled in the home page of the website, then the priority is high and severity is low to fix it.

**42) how to estimate test cases?**

List of Software Test Estimation Techniques

* Work Breakdown Structure
* 3-Point Software Testing Estimation Technique
* Wideband Delphi technique
* Function Point/Testing Point Analysis
* Use – Case Point Method
* Percentage distribution
* Ad-hoc method.

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

Some times you just don’t pay proper attention what the company-defined processes are and these are for what purposes. There are some myths in testers that they should only go with company processes even these processes are not applicable for their current testing scenario. This results in incomplete and inappropriate application testing.

**44) what are test design techniques?**

By **design** we mean to create a plan for how to implement an idea and **technique** is a method or way for performing a task. So, **Test Design** is creating a set of inputs for given software that will provide a set of expected outputs. ... Broadly speaking there are two main categories of **Test Design Techniques**.

45) if we dont have time to test call test cases what we will do?

**1) Incorrect Estimation**:

If you started with an inaccurate expectation, things are bound to fail. A good test estimate must take the following into account:

* **Time for preparatory tasks –** We are talking about tasks such as:
  + Identifying and putting together a regression suite
  + Creating Test data
  + Time to determine test readiness (E.g.: Smoke/Sanity Test), etc.
* **Test case maintenance**: Test cases are long-term usage assets. They are sure to undergo minor updates during execution. It is recommended that for new products up to 30% of your test execution time should be allocated for these minor maintenance tasks. All teams and projects might not need 30%, but do allocate some time and effort for this task.
* [**Ad-hoc**](http://www.softwaretestinghelp.com/ad-hoc-testing/)**/Exploratory testing**– The count of scripted tests is a major denominator for test estimation numbers. However, no test team in this world will deny exploring your software even if the model is dominantly scripted.
* **Reporting/Communication**– This includes triage/stand up meetings, updating work management tools etc.
* **Contingency factor:** Standards recommend 25-30% buffer to your original estimates. But teams can rarely afford it. Even then, leave a little breathing room, when possible.
* **Team and its capabilities:** If you have a new team or if they are using a tool for the first time, you might need to set some time aside for training. Tailor your estimates based on your team you are working with.

**Unstable builds and other technical problems:**

* **Smoke/Sanity test failure**: When the basic tests on the AUT fail after deployment into QA environment there is pretty much nothing the QA team can do towards test execution. It is true that we can work on other tasks while this happens, but it still will not fill the [test cycle](http://www.softwaretestinghelp.com/what-is-software-testing-life-cycle-stlc/) time. So, this is a major contributor to time wasted.
* [**Test data**](http://www.softwaretestinghelp.com/tips-to-design-test-data-before-executing-your-test-cases/)**unavailable**: Production-like data is a must for every testing project. Not getting this into the QA environment on time is also another blocking factor. Sometimes testers can work around this by [creating and managing their own test data](http://www.softwaretestinghelp.com/test-data-management-techniques/), but it is time-consuming and might not always be on-point.
* [**Environment issues**](http://www.softwaretestinghelp.com/test-bed-test-environment-management-best-practices/)– The build failing deployments, the server keeps getting timed out, many more such issues eat away your test cycle. This probably stems from the fact that, some companies (not all) undermine the importance of a good, live-like environment for effective QA. They often try to get away low-capacity servers and make-do set ups. This is really a short-time fix and does nobody any favors. In fact, it could cost them the quality of testing and loss of valuable test time.

**45) How we learn the functionality of the system?**

A Learning Management System (LMS) is a software application that is used to plan, deliver, publish and place self-paced online courses in online catalogs. Employees login to the LMS by using a browser. They can select courses from the online catalog and begin their online study. Alternatively, a manager can assign courses to employees based on their learning needs.

An LMS allows managers and administrates to track course completions, current status or performance of employees. In fact, all employee activities in the LMS can be tracked that could be useful for performance evaluation, competency management and other related functions.

46)what are the tools to manage defects/stories?

It is inevitable that at some point in time every software application will contain Defects. Non-realistic teams deny this situation and discover that they were wrong. Realistic teams accept this fact and focus on wondering when these Defects are caused, how long it takes before they are found, and even more important when they are resolved. Many studies have proved that fixing a Defect found in a production stage is much more expensive (in terms of cost and effort) than fixing it when found in earlier stages of the project (NIST study). These same studies concluded that “software bugs, or Defects, cost the US economy an estimated $59 billion annually, or about 0.6 percent of the gross domestic product”. Another study at the IBM Systems Sciences Institute states that the cost of finding & fixing a Defect in the maintenance stage is 100 times higher than when it was found.

47) who will assign the work?

Typically, once tasks are [entered into a schedule](https://support.office.com/en-us/article/entered-into-a-schedule-6990803e-d0c7-48aa-94c9-ef14387f6792), people are assigned to work on them. After all, projects don’t get done by themselves.

1. In the Gantt Chart, go to the **Resource Names** column.
2. Click the arrow and pick the people to work on the task.  
   
3. If you don’t see the name of the person you want to assign, enter a new name.

If you don’t see any names in the list, then no one has been added to the project yet. Either enter a new name in the column, or [add resources to your project](https://support.office.com/en-us/article/add-resources-to-your-project-af9bc1f7-41cf-40ec-a497-c8694fc7f844) before assigning them to work on tasks.

To add resources like equipment and materials, see [Add and assign material resources to tasks](http://ntrajkovski.wordpress.com/2011/11/07/assigning-material-and-cost-resources-in-ms-project-2010/).

**In Project Professional 2016, some resources may require approval from a resource manager.** If you're connected to Project Online, there may be some resources that require you request a resource engagement before the resource can be used on your project. Project shows this indicator Engagement required if you need to request an engagement. An engagement is an agreement between a project manager and a resource manager, to allow a specific resource to work on a project for a certain period of time, at a certain percentage of that resource's time, or for a certain number of hours.

48) what is requirement traceability matrix?

The **Requirements Traceability Matrix** (RTM) is a document that links **requirements**throughout the validation process. The purpose of the **Requirements Traceability Matrix** is to ensure that all **requirements** defined for a system are tested in the test protocols.

49) what are typical environments we have in projects?

To ensure that you have a sound grasp of the basic principles of project management we will now review some key issues. Firstly, a project is a time bounded activity which is usually a one-off rather than an ongoing continuous process. The project environment in which the project takes place includes such factors as geography, the economy, government, resources, sub-contractors, suppliers, national culture, organisational strategy, other projects, competitors, customers and users. All these factors have a bearing on the likely success or otherwise of the project. The project definition involves identifying the end state that project management is trying to achieve, the range of the responsibilities taken on by the project and the strategy of how the objectives are going to be met. The objectives need to be clear, measurable and quantifiable. The scope of the project is a boundary setting exercise which attempts to define the dividing line between what the project will do and what it won’t do. The scope of the project is formalized in the project specification and the strategy defines the general approach to the project. Finally key milestones must be identified to enable reviews of quality, cost and time issues.

50) what are different defect metrics and measurements we prepare in testing

*A Metric is a quantitative measure of the degree to which a system, system component, or process possesses a given attribute.*

***Metrics can be defined as “STANDARDS OF MEASUREMENT”.***

Software Metrics are used to measure the quality of the project. Simply, Metric is a unit used for describing an attribute. Metric is a scale for measurement.

Suppose, in general, “Kilogram” is a metric for measuring the attribute “Weight”. Similarly, in software, “How many issues are found in thousand lines of code?”, h*ere No. of issues is one measurement & No. of lines of code is another measurement. Metric is defined from these two measurements*.

51) 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**.

52) **What is** **QA 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.

53) 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.

**54)** what is production environment?

A **production environment** is where the real-time staging of programs that run an organization are executed, and includes the personnel, processes, data, hardware, and software needed to perform day-to-day operations.

**55)** how to deal the production defects?

First, teams must have business representation to help them make priority decisions.  The most common way to do this, is to designate someone as a Product Manager.  This could be a team member who communicates regularly with business stakeholders.  Ideally, this is a business person, who works directly with the team.

Once a defect has been spotted (defect according to our criteria above) the team needs to make that visible.  In the case of a Scrum team, this means logging this in your product backlog, as you would a story.   For a team using Kanban, they would get that defect into their parking lot.  This also means, the team gives business the opportunity to prioritize the defect work.

The exception is for critical production defects.  This is covered in a section below.  For all other defects, once they are in some type of backlog, then the team needs to plan when to handle these defects.

One question that might come up is should we estimate the effort for these defects.  Opinions differ [2], but to give business an idea on effort for those defects, it makes sense to size them in some way.  If your team is used to story points, then utilize that method.

For teams using Scrum, tasking could be used.  In other words, when the team is in planning session, they will create the task time estimate for a defect.  For the dev tasks, they may estimate, 10 hours, for testing time, 4 hours, and now the team has an estimate that can be used with the teams total time budget to help identify if their sprint commitment is reasonable.

- See more at: http://www.devx.com/architect/Article/46310#sthash.PtuakP2u.dpuf